Package 'brittnu'

March 27, 2021

Title Britt's Nu for Dirichlet-Distributed Data

Version 0.1.0

Description This function computes the Britt's nu reliability coefficient for data sets composed of multiple Dirichlet-distributed deviates with at least two categories. This is especially useful to assess the reliability of cross-validations of latent Dirichlet allocation.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Imports utils,
plyr,
topicmodels,
stats,
methods

R topics documented:

brittnu *Britt's Nu for Dirichlet-Distributed Data*

Description

This function computes the Britt's nu reliability coefficient for data sets composed of multiple Dirichlet-distributed deviates, as described by Britt (under review).

Usage

```
brittnu(
  x,
  type = NA,
  alpha = NA,
  estimate_pairwise_alpha_from_joint = TRUE,
  symmetric_alpha = FALSE,
  shuffle = FALSE,
  method = "rotational",
  different_documents = FALSE,
  pairwise = TRUE,
  samples = 1000,
  lower_bound = FALSE,
  convcrit = 1e-05,
  maxit = 1000,
  verbose = FALSE,
  zero = (10^(-16)),
  zero2 = (10^(-255))
)
```

Arguments

x	A list with each entry being a single cross-validation iteration; each list entry should be either the output from a LDA function call or a 2D double vector with distributions as the rows and each category within each distribution as the columns (such that each row in the vector sums to 1)
type	A string indicating the type of reliability being assessed (either word-topic or topic-document); if x contains the output from multiple LDA function calls, this must be set to either "wt" (word-topic reliability) or "td" (topic-document reliability)
alpha	A list whose length is equal to the number of distributions, with each list element containing a vector of length K comprising the concentration parameters for the K categories in that distribution; if this is NA, the concentration parameters will be estimated from x
estimate_pairwise_alpha_from_joint	A boolean value indicating, when estimating reliability for pairs of cross-validation iterations, whether each pair of iterations should use the concentration parameters estimated across all iterations (TRUE) or whether separate sets of concentration parameters should be estimated for each pair of cross-validation iterations (FALSE); this parameter only has an effect when alpha=NA, and it is strongly suggested that the default value of TRUE be used for this parameter

`symmetric_alpha`
: A boolean value indicating whether all concentration parameters in each cross-validation iteration should be assumed to be equal, which is common in LDA; this parameter only has an effect when `alpha=NA`

`shuffle`
: A boolean value indicating whether or not the topics in each iteration may have been shuffled in the context of LDA; if this is `TRUE` and `type="wt"` or `type=NA`, then the distributions within each iteration will be reordered to achieve a (local) optimal fit, whereas if this is `TRUE` and `type="td"`, the categories within each distribution will be reordered instead, so if you want the categories within each distribution to be reordered, then you should set `type="td"` even if `x` contains raw data rather than the output from multiple `LDA` function calls

`method`
: A string indicating what method (`"rotational"` or `"forward"`) will be used to reorder topics (see Britt, under review); the `"forward"` method will be implemented at a later date

`different_documents`
: A boolean value indicating, if each list element in `x` represents the output from a `LDA` function call, whether some cross-validation iterations used different sets of documents than others

`pairwise`
: A boolean value indicating whether pairwise reliability between individual cross-validation iterations should be computed; if `FALSE`, pairwise reliability will be ignored in order to reduce the time and memory complexity of the computation

`samples`
: An integer value indicating how many samples to use to estimate expected differences when `shuffle=TRUE`

`lower_bound`
: A boolean value indicating whether the lower bound of the expected differences (based on i.i.d. beta distributions) should be used rather than shuffling Dirichlet distributions to estimate the exact difference in every sample when `shuffle=TRUE`; for Dirichlet distributions with many categories and function calls with a large value of samples, setting this to `TRUE` is strongly recommended, as reordering numerous samples with a large number of categories may require an unreasonable amount of time

`convcrit`
: A numeric value indicating the threshold for convergence used to estimate concentration parameters when `alpha=NA`

`maxit`
: A numeric value indicating the maximum number of iterations used to estimate concentration parameters when `alpha=NA`

`verbose`
: A boolean value indicating whether `brittnu()` and its helper functions should provide progress updates

`zero`
: A numeric value; if `type="wt"` and `different_documents=TRUE`, then whenever a word appears in some cross-validation iterations but not others, in order to prevent errors, this value is assigned as its allocation to all topics within any cross- validation iteration in which the word did not originally appear

`zero2`
: A numeric value; if `alpha=NA`, then in order to prevent errors when concentration parameters are being estimated, this is the minimum allocation permitted for any given category during the estimation procedure

Details

When estimating Britt's nu, it is common to compute four separate coefficients in this family: Britt's single-distribution nu for all cross-validation iterations, Britt's single-distribution nu for pairs of iterations, Britt's omnibus nu for all cross-validation iterations, and Britt's single-distribution nu

for pairs of iterations. The versions that use all cross-validation iterations are more vital to the assessment of reliability, but the pairwise versions are sometimes useful as diagnostic tools.

By default, `brittnu()` provides all four versions of Britt's nu. Since computing the pairwise versions of Britt's nu requires more time than using all cross-validation iterations, the pairwise versions may optionally be omitted from the computation by specifying `pairwise = FALSE`.

The expected difference component of Britt's nu is estimated using a large number of samples from Dirichlet distributions with the same concentration parameters as the original data set. In some cases, this process can become computationally intensive. For those instances, it is advisable to set `lower_bound = TRUE`, which uses i.i.d. beta distributions to estimate the lower bound of the expected differences between observations, ultimately yielding an estimated lower bound for Britt's nu itself.

One of the most important uses for Britt's nu is to assess the reliability of the allocations of words to topics or of topics to documents via latent Dirichlet allocation (LDA). When the results of multiple LDA cross-validation iterations are provided as `x`, the `type` parameter must be set to either `"wt"` or `"td"` to indicate whether word-topic or topic-document reliability, respectively, should be assessed.

Crucially, different LDA cross-validation iterations may result in the same topics appearing in a different sequence. As such, those topics may need to be reordered in order to yield an optimal fit. In practice, it is generally infeasible to assess every possible combination of topic across all cross-validation iterations. As such, `brittnu()` provides two methods of converging toward an optimal topic order for each iteration. The default, `method = "rotational"`, takes the provided sequence and swaps pairs of categories until no further swaps would further improve the fit. Alternatively, `method = "forward"` (in development) assigns the best-fitting category, one at a time, until all categories have been assigned to all distributions. The forward method is more computationally efficient for use cases with many categories and cross-validation iterations, but it is likely to generate a worse fit than the rotational method. On the other hand, the greater reliability is, the better the forward method is expected to perform, and when reliability is poor, it may not be important to precisely estimate Britt's nu.

Additional usage notes:

1. When importing your own data set (rather than a list of objects outputted from `LDA`), each row of the data set should be a distribution (e.g., an LDA topic), and each column should be a category within that distribution (e.g., a word).

2. It is generally recommended that `estimate_pairwise_alpha_from_joint` be set to `TRUE`, which is the default setting. If concentration parameters must be estimated from the data, and if all cross-validation iterations used the same data set or are otherwise assumed to have emerged from the same underlying distribution, then there is little reason to expect different sets of concentration parameters to be necessary across iterations. You should only consider setting `estimate_pairwise_alpha_from_joint` to `FALSE` if different data sets that may not have come from the same underlying distribution were used for different cross-validation iterations.

3. If you are assessing the reliability of the allocations of words to topics set of documents. For instance, if `mydata1`, `mydata2`, `mydata3`, `mydata4`, and `mydata5` each contain 20 documents from a single data set, then you could run

```
cv1 <- LDA(mydata1, k=15)
cv2 <- LDA(mydata2, k=15)
cv3 <- LDA(mydata3, k=15)
cv4 <- LDA(mydata4, k=15)
cv5 <- LDA(mydata5, k=15)
cv <- list(cv1, cv2, cv3, cv4, cv5)
rel <- brittnu(cv, type="wt", shuffle=TRUE, different_documents=TRUE)
print(rel)
```

to assess the word-topic allocation reliability. Any words that appear in some cross-validation iterations but not others will automatically be set to an allocation near 0 for any iterations in which they are absent from the data set. (In other words, if each topic in that iteration represents its own Dirichlet distribution, the missing word will be treated as a category with a concentration parameter and allocation of zero.) If this applies in your case, set `different_documents = TRUE` when conducting your LDA. Note, however, that if you are assessing the reliability of the allocations of topics to documents, then each document represents its own respective Dirichlet distribution, and any LDA in which a given document does not appear will simply be excluded from the reliability computation for that document, such that there is little benefit in using different sets of documents for different cross-validation iterations. Therefore, when assessing topic-document reliability, it is advised that all documents be included in all cross-validation iterations.

4. For Dirichlet distributions with many categories and function calls with a large value for the samples parameter, setting `lower_bound=TRUE` is strongly recommended, as shuffling numerous samples with a large number of categories may require an unreasonable amount of time. Additionally, for Dirichlet distributions with many categories, precise estimates of the concentration parameters may require excessive time, so when possible, a priori known concentration parameters should be provided via the `alpha` parameter rather than estimating them from the data. When this is not possible, it may be advisable to change `convcrit` from its default value (0.00001) to a larger number. (With approximately 10,000 categories, a single iteration of the convergence algorithm to estimate concentration parameters may take several minutes to complete.) You may also wish to set `verbose = TRUE` to receive periodic progress updates and ensure that the computation is proceeding as expected.

Value

A list with four elements: Britt's single-distribution nu for all cross-validation iterations (as a numeric vector of length K indicating the reliability for each of the K categories), Britt's single-distribution nu for pairs of cross-validation iterations (as a list containing lists containing numeric vectors of length K, e.g., the second element of the first list contains the reliability for cross-validation iterations 1 and 2), Britt's omnibus nu for all cross-validation iterations (as a numeric value), and Britt's omnibus nu for pairs of cross-validation iterations (as a list containing lists containing numeric values, e.g., the second element of the first list contains the reliability for cross-validation iterations 1 and 2)

References

Britt, B. C. (under review). An interrater reliability coefficient for beta-distributed and Dirichlet-distributed data.

Examples

```
require(topicmodels)
data("AssociatedPress")
set.seed(1797)
ap1 <-  LDA(AssociatedPress, k = 40)
set.seed(1798)
ap2 <-  LDA(AssociatedPress, k = 40)
set.seed(1799)
ap3 <-  LDA(AssociatedPress, k = 40)
set.seed(1800)
ap4 <-  LDA(AssociatedPress, k = 40)
set.seed(1801)
ap5 <-  LDA(AssociatedPress, k = 40)
set.seed(1802)
```

```
ap6 <-  LDA(AssociatedPress, k = 40)
set.seed(1803)
ap7 <-  LDA(AssociatedPress, k = 40)
set.seed(1804)
ap8 <-  LDA(AssociatedPress, k = 40)
set.seed(1805)
ap9 <-  LDA(AssociatedPress, k = 40)
set.seed(1806)
ap10 <-  LDA(AssociatedPress, k = 40)
set.seed(1807)
ap11 <-  LDA(AssociatedPress, k = 40)
set.seed(1808)
ap12 <-  LDA(AssociatedPress, k = 40)
set.seed(1809)
ap13 <-  LDA(AssociatedPress, k = 40)
set.seed(1810)
ap14 <-  LDA(AssociatedPress, k = 40)
set.seed(1811)
ap15 <-  LDA(AssociatedPress, k = 40)
set.seed(1812)
ap16 <-  LDA(AssociatedPress, k = 40)
set.seed(1813)
ap17 <-  LDA(AssociatedPress, k = 40)
set.seed(1814)
ap18 <-  LDA(AssociatedPress, k = 40)
set.seed(1815)
ap19 <-  LDA(AssociatedPress, k = 40)
set.seed(1816)
ap20 <-  LDA(AssociatedPress, k = 40)
ap_all_40 <- list(ap1,ap2,ap3,ap4,ap5,ap6,ap7,ap8,ap9,ap10,ap11,ap12,ap13,
                  ap14,ap15,ap16,ap17,ap18,ap19,ap20)
set.seed(1817)
reliability_40_td <- brittnu(ap_all_40, type="td", symmetric_alpha=TRUE,
                             shuffle=TRUE, samples=1000)
#Britt's single-distribution nu for all iterations
print(reliability_40_td[[1]])
#Britt's single-distribution nu for pairs of iterations
print(reliability_40_td[[2]])
#Britt's omnibus nu for all iterations
print(reliability_40_td[[3]])
#Britt's omnibus nu for pairs of iterations
print(reliability_40_td[[4]])
#All Britt's nu values for topic-document reliability
print(reliability_40_td)
set.seed(1820)
reliability_40_wt <- brittnu(ap_all_40, type="wt", symmetric_alpha=TRUE,
                             shuffle=TRUE, samples=1000)
#Britt's single-distribution nu for all iterations
print(reliability_40_wt[[1]])
#Britt's single-distribution nu for pairs of iterations
print(reliability_40_wt[[2]])
#Britt's omnibus nu for all iterations
print(reliability_40_wt[[3]])
#Britt's omnibus nu for pairs of iterations
print(reliability_40_wt[[4]])
#All Britt's nu values for word-topic reliability
print(reliability_40_wt)
```

`compute_standard_single_expectation` *Compute Standard Single Expectation*

Description

This helper function computes the expected squared difference between deviates from the same underlying Dirichlet distribution when the categories in each distribution are not allowed to be reordered.

Usage

```
compute_standard_single_expectation(
  joint_alpha,
  pairwise_alpha,
  pairwise,
  verbose = FALSE
)
```

Arguments

`joint_alpha`	A vector of concentration parameters representing all cross-validation iterations
`pairwise_alpha`	A list of lists of vectors of concentration parameters representing all pairs of cross-validation iterations
`pairwise`	A boolean value indicating whether pairwise reliability between individual cross-validation iterations should be computed; if `FALSE`, pairwise reliability will be ignored in order to reduce the time and memory complexity of the computation
`verbose`	A boolean value indicating whether progress updates should be provided

Value

A two-element list indicating the expected difference between reordered Dirichlet deviates based on the concentration parameters given in `joint_alpha` and `pairwise_alpha`, respectively

`dirichlet.mle` *Dirichlet.MLE*

Description

This helper function estimates the concentration parameters of the Dirichlet distribution underlying multiple deviates, with no restrictions placed on the values of those concentration parameters. This function is heavily based on `dirichlet.mle`, with modifications to avoid potential singularities in the estimation procedure.

Usage

```
dirichlet.mle(
  x,
  weights = NULL,
  eps = 10^(-5),
  convcrit = 1e-05,
  maxit = 1000,
  oldfac = 0.3,
  verbose = FALSE
)
```

Arguments

x	A list with each entry being a single cross-validation iteration; each list entry should be either the output from a LDA function call or a 2D double vector with distributions as the rows and each category within each distribution as the columns (such that each row in the vector sums to 1)
weights	A numeric vector used to calibrate the initial estimates of concentration parameters
eps	A numeric value used as a tolerance parameter to prevent logarithms of zero
convcrit	A numeric value indicating the threshold for convergence used to estimate concentration parameters
maxit	A numeric value indicating the maximum number of iterations used to estimate concentration parameters
oldfac	A numeric value between 0 and 1 used as the convergence acceleration factor
verbose	A boolean value indicating whether progress updates should be provided

Value

A list of the estimated concentration parameters, the sum of those estimated concentration parameters, and the ratio between each estimated concentration parameter and the sum of those parameters

---

estimate_alpha_from_data

Estimate Alpha from Data

Description

This helper function estimates the expected concentration parameters of the Dirichlet distribution underlying multiple deviates.

Usage

```
estimate_alpha_from_data(
  x,
  pairwise,
  estimate_pairwise_alpha_from_joint,
  symmetric_alpha = FALSE,
  convcrit = 1e-05,
```

```
  maxit = 1000,
  verbose = FALSE
)
```

Arguments

x	A list with each entry being a single cross-validation iteration; each list entry should be either the output from a LDA function call or a 2D double vector with distributions as the rows and each category within each distribution as the columns (such that each row in the vector sums to 1)
pairwise	A boolean value indicating whether pairwise reliability between individual cross-validation iterations should be computed; if FALSE, pairwise reliability will be ignored in order to reduce the time and memory complexity of the computation
estimate_pairwise_alpha_from_joint	A boolean value indicating, when estimating reliability for pairs of cross-validation iterations, whether each pair of iterations should use the concentration parameters estimated across all iterations (TRUE) or whether separate sets of concentration parameters should be estimated for each pair of cross-validation iterations (FALSE); it is strongly suggested that the default value of TRUE be used for this parameter
symmetric_alpha	A boolean value indicating whether all concentration parameters in each cross-validation iteration should be assumed to be equal, which is common in LDA
convcrit	A numeric value indicating the threshold for convergence used to estimate concentration parameters
maxit	A numeric value indicating the maximum number of iterations used to estimate concentration parameters
verbose	A boolean value indicating whether progress updates should be provided

Value

A two-element list indicating the estimated concentration parameters for all concentration parameters and for each pair of cross-validation iterations, respectively

---

estimate_shuffled_single_expectation

Estimate Shuffled Single Expectation

Description

This helper function estimates the expected squared difference between deviates from the same underlying Dirichlet distribution when the categories in each distribution are allowed to be reordered.

Usage

```
estimate_shuffled_single_expectation(
  joint_alpha,
  pairwise_alpha,
  pairwise,
  samples = 1000,
```

```
  lower_bound = TRUE,
  method = "rotational",
  num_iter = (length(pairwise_alpha) + 1),
  zero2 = (10^(-255)),
  verbose = FALSE
)
```

Arguments

joint_alpha	A vector of concentration parameters representing all cross-validation iterations
pairwise_alpha	A list of lists of vectors of concentration parameters representing all pairs of cross-validation iterations
pairwise	A boolean value indicating whether pairwise reliability between individual cross-validation iterations should be computed; if FALSE, pairwise reliability will be ignored in order to reduce the time and memory complexity of the computation
samples	An integer value indicating how many samples to use to estimate expected differences when shuffle=TRUE
lower_bound	A boolean value indicating whether the lower bound of the expected differences (based on i.i.d. beta distributions) should be used rather than shuffling Dirichlet distributions to estimate the exact difference in every sample when shuffle=TRUE; for Dirichlet distributions with many categories and function calls with a large value of samples, setting this to TRUE is strongly recommended, as reordering numerous samples with a large number of categories may require an unreasonable amount of time
method	A string indicating what method ("rotational" or "forward") will be used to reorder topics (see Britt, under review); the "forward" method will be implemented at a later date
num_iter	An integer value indicating the number of cross-validation iterations
zero2	A numeric value; in order to prevent errors when generating Dirichlet deviates, any concentration parameters that are less than zero2 are changed to be zero2
verbose	A boolean value indicating whether progress updates should be provided

Value

A two-element list indicating the expected difference between reordered Dirichlet deviates based on the concentration parameters given in joint_alpha and pairwise_alpha, respectively

---

generate_dirichlet_deviate_diffs

Generate Squared Dirichlet Deviate Differences

Description

This helper function generates two deviates from the same underlying Dirichlet distribution and returns the sum of the squared differences between the categories of those deviates.

Usage

```
generate_dirichlet_deviate_diffs(x, samples, zero2 = (10^(-255)))
```

Arguments

x	The concentration parameters used to generate Dirichlet deviates
samples	An integer value indicating how many samples to use to compute the sum of squared differences
zero2	A numeric value; in order to prevent errors when generating Dirichlet deviates, any concentration parameters that are less than zero2 are changed to be zero2
method	A string indicating what method ("rotational" or "forward") will be used to reorder topics (see Britt, under review); the "forward" method will be implemented at a later date

Value

A numeric value representing the sum of squared differences between Dirichlet deviates for samples times the number of pairs of deviates

---

rdirichlet *RDirichlet*

Description

This helper function generates a set of deviates from a Dirichlet distribution with underlying concentration parameters equal to alpha. This function is heavily based on rdirichlet, with modifications to prevent concentration parameters equal to 0.

Usage

```
rdirichlet(alpha, n, zero2 = (10^(-255)))
```

Arguments

alpha	A numeric vector indicating the concentration parameters of the Dirichlet distribution from which deviates should be generated
n	A numeric value indicating the number of Dirichlet deviates to be generated
zero2	A numeric value representing the minimum allocation permitted for any given Dirichlet category in order to prevent errors

Value

A numeric vector with n Dirichlet deviates

shuffle_distributions *Shuffle Distributions*

Description

This helper function reorders the categories in multiple Dirichlet deviates in order to minimize the sum of squared differences between them.

Usage

```
shuffle_distributions(x, method = "rotational", verbose = FALSE)
```

Arguments

x	A list with each entry being a single cross-validation iteration; each list entry should be either the output from a LDA function call or a 2D double vector with distributions as the rows and each category within each distribution as the columns (such that each row in the vector sums to 1)
method	A string indicating what method ("rotational" or "forward") will be used to reorder topics (see Britt, under review); the "forward" method will be implemented at a later date
verbose	A boolean value indicating whether progress updates should be provided

Value

A two-element list containing the reordered data set and a matrix indicating the relationship between each reordered category and its orginal position in the data

sirt_digamma1 *Sirt Digamma 1*

Description

This helper function estimates the derivative of the digamma function. This function is directly drawn from sirt_digamma1, as published at https://github.com/cran/sirt/blob/master/R/sirt_digamma1.R.

Usage

```
sirt_digamma1(x, h = 0.001)
```

Arguments

x	A vector of concentration parameters
h	A numeric value used to define a pair of nearby observations

Value

A list of the estimated concentration parameters, the sum of those estimated concentration parameters, and the ratio between each estimated concentration parameter and the sum of those parameters

`symmetric.dirichlet.mle` *Symmetric.Dirichlet.MLE*

Description

This helper function estimates the concentration parameters of the Dirichlet distribution underlying multiple deviates, assuming that those concentration parameters are all equal. This function is heavily based on `dirichlet.mle`, with modifications to restrict all concentration parameters to be equal and to avoid potential singularities in the estimation procedure.

Usage

```
symmetric.dirichlet.mle(
  x,
  weights = NULL,
  eps = 10^(-5),
  convcrit = 1e-05,
  maxit = 1000,
  oldfac = 0.3,
  verbose = FALSE
)
```

Arguments

`x`	A list with each entry being a single cross-validation iteration; each list entry should be either the output from a `LDA` function call or a 2D double vector with distributions as the rows and each category within each distribution as the columns (such that each row in the vector sums to 1)
`weights`	A numeric vector used to calibrate the initial estimates of concentration parameters
`eps`	A numeric value used as a tolerance parameter to prevent logarithms of zero
`convcrit`	A numeric value indicating the threshold for convergence used to estimate concentration parameters
`maxit`	A numeric value indicating the maximum number of iterations used to estimate concentration parameters
`oldfac`	A numeric value between 0 and 1 used as the convergence acceleration factor
`verbose`	A boolean value indicating whether progress updates should be provided

Value

A list of the estimated concentration parameters, the sum of those estimated concentration parameters, and the ratio between each estimated concentration parameter and the sum of those parameters

Index