Package ‘rainette’

June 25, 2021

Type Package
Title The Reinert Method for Textual Data Clustering
Version 0.2.0
Date 2021-06-25
Maintainer Julien Barnier <julien.barnier@cnrs.fr>
Description An R implementation of the Reinert text clustering method. For more details about the algorithm see the included vignettes or Reinert (1990) <doi:10.1177/075910639002600103>.
License GPL (>= 3)
VignetteBuilder knitr

URL https://juba.github.io/rainette/

BugReports https://github.com/juba/rainette/issues

Encoding UTF-8

Imports dplyr (>= 1.0.0), tidyr, purrr, ggplot2, stringr, quanteda (>= 2.1), quanteda.textstats, RSpectra, dendextend, ggbwordcloud, gridExtra, rlang, shiny, miniUI, formatR, highr, progressr, Rcpp (>= 1.0.3)
Suggests testthat, knitr, rmarkdown, tm, FNN, quanteda.textmodels

RoxygenNote 7.1.1

LinkingTo Rcpp

NeedsCompilation yes

Author Julien Barnier [aut, cre],
Florian Privé [ctb]

Repository CRAN

Date/Publication 2021-06-25 14:40:02 UTC
clusters_by_doc_table

Returns the number of segment of each cluster for each source document

Description

Returns the number of segment of each cluster for each source document

Usage

clusters_by_doc_table(obj, clust_var = NULL, doc_id = NULL, prop = FALSE)

Arguments

obj a corpus, tokens or dtm object
clust_var name of the docvar with the clusters
doc_id docvar identifying the source document
prop if TRUE, returns the percentage of each cluster by document

Details

This function is only useful for previously segmented corpus. If doc_id is NULL and there is a segment_source docvar, it will be used instead.
See Also

docs_by_cluster_table()

Examples

```r
require(quanteda)
corpus <- data_corpus_inaugural
corpus <- head(corpus, n = 10)
corpus <- split_segments(corpus)
tok <- tokens(corpus, remove_punct = TRUE)
tok <- tokens_remove(tok, stopwords("en"))
dtm <- dfm(tok, tolower = TRUE)
dtm <- dfm_trim(dtm, min_docfreq = 2)
res <- rainette(dtm, k = 3, min_segment_size = 15)
corpus$cluster <- cutree(res, k = 3)
clusters_by_doc_table(corpus, clust_var = "cluster", prop = TRUE)
```

### cluster_tab

*Split a dtm into two clusters with reinert algorithm*

**Description**

Split a dtm into two clusters with reinert algorithm

**Usage**

```r
cluster_tab(dtm, cc_test = 0.3, tsj = 3)
```

**Arguments**

- `dtm` to be split, passed by rainette
- `cc_test` maximum contingency coefficient value for the feature to be kept in both groups.
- `tsj` minimum feature frequency in the dtm

**Details**

Internal function, not to be used directly

**Value**

An object of class hclust and rainette
**cutree**  
*Cut a tree into groups*

**Description**
Cut a tree into groups

**Usage**
cutree(tree, ...)

**Arguments**
- **tree**  
  the hclust tree object to be cut
- ...  
  arguments passed to other methods

**Details**
If tree is of class rainette, invokes cutree_rainette(). Otherwise, just run stats::cutree().

**Value**
A vector with group membership.

---

**cutree_rainette**  
*Cut a rainette result tree into groups of documents*

**Description**
Cut a rainette result tree into groups of documents

**Usage**
cutree_rainette(hres, k = NULL, h = NULL, ...)

**Arguments**
- **hres**  
  the rainette result object to be cut
- **k**  
  the desired number of clusters
- **h**  
  unsupported
- ...  
  arguments passed to other methods

**Value**
A vector with group membership.
**Description**

Cut a rainette2 result object into groups of documents

**Usage**

```r
cutree_rainette2(res, k, criterion = c("chi2", "n"), ...)
```

**Arguments**

- `res`: the rainette2 result object to be cut
- `k`: the desired number of clusters
- `criterion`: criterion to use to choose the best partition. `chi2` means the partition with the maximum sum of chi2, `n` the partition with the maximum size.
- `...`: arguments passed to other methods

**Value**

A vector with group membership.

**See Also**

- `rainette2_complete_groups()`

---

**docs_by_cluster_table**

*Returns, for each cluster, the number of source documents with at least n segments of this cluster*

**Description**

Returns, for each cluster, the number of source documents with at least n segments of this cluster

**Usage**

```r
docs_by_cluster_table(obj, clust_var = NULL, doc_id = NULL, threshold = 1)
```

**Arguments**

- `obj`: a corpus, tokens or dtm object
- `clust_var`: name of the docvar with the clusters
- `doc_id`: docvar identifying the source document
- `threshold`: the minimal number of segments of a given cluster that a document must include to be counted
Details

This function is only useful for previously segmented corpus. If doc_id is NULL and there is a
segment_source docvar, it will be used instead.

See Also

clusters_by_doc_table()

Examples

```r
require(quanteda)
corpus <- data_corpus_inaugural
corpus <- head(corpus, n = 10)
corpus <- split_segments(corpus)
tok <- tokens(corpus, remove_punct = TRUE)
tok <- tokens_remove(tok, stopwords("en"))
dtm <- dfm(tok, tolower = TRUE)
dtm <- dfm_trim(dtm, min_docfreq = 2)
res <- rainette(dtm, k = 3, min_segment_size = 15)
corpus$cluster <- cutree(res, k = 3)
docs_by_cluster_table(corpus, clust_var = "cluster")
```

---

**import_corpus_iramuteq**

*Import a corpus in Iramuteq format*

Description

Import a corpus in Iramuteq format

Usage

```
import_corpus_iramuteq(f, id_var = NULL, thematics = c("remove", "split"), ...)
```

Arguments

- `f` a file name or a connection
- `id_var` name of metadata variable to be used as documents id
- `thematics` if "remove", thematics lines are removed. If "split", texts as splitted at each
thematic, and metadata duplicated accordingly
- `...` arguments passed to file if f is a file name.

Details

A description of the Iramuteq corpus format can be found here: [http://www.iramuteq.org/documentation/html/2-2-2-les-regles-de-formatages](http://www.iramuteq.org/documentation/html/2-2-2-les-regles-de-formatages)
merge_segments

Value
A quanteda corpus object. Note that metadata variables in docvars are all imported as characters.

merge_segments Merges uces into uc according to minimum uc size

Description
rainette_uc_index docvar

Usage
merge_segments(dtm, min_segment_size = 10, doc_id = NULL)

Arguments
dtm dtm of uces, with a rainette_uce_id docvar
min_segment_size minimum number of forms by uc
doc_id character name of a dtm docvar which identifies source documents.

Details
If min_segment_size == 0, different uc ids are added to the dtm docvars (i.e., no uce are merged together). If min_segment_size > 0 then doc_id must be provided unless the corpus comes from split_segments, in this case segment_source is used by default.

Value
the original dtm with a new rainette_uc_id docvar.

order_docs return documents indices ordered by CA first axis coordinates

Description
return documents indices ordered by CA first axis coordinates

Usage
order_docs(m)

Arguments
m dtm on which to compute the CA and order documents, converted to an integer matrix.
Details

Internal function, not to be used directly

Value

ordered list of document indices

---

**rainette**

*Corpus clustering based on the Reinert method - Simple clustering*

---

Description

Corpus clustering based on the Reinert method - Simple clustering

Usage

```r
rainette(
  dtm,
  k = 10,
  min_segment_size = 0,
  doc_id = NULL,
  min_split_members = 5,
  cc_test = 0.3,
  tsj = 3,
  min_members,
  min_uc_size
)
```

Arguments

dtm         quanteda dfm object of documents to cluster, usually the result of `split_segments()`
k           maximum number of clusters to compute
min_segment_size  minimum number of forms by document
doc_id       character name of a dtm docvar which identifies source documents.
min_split_members  don’t try to split groups with fewer members
cc_test      contingency coefficient value for feature selection
tsj          minimum frequency value for feature selection
min_members  deprecated, use `min_split_members` instead
min_uc_size  deprecated, use `min_segment_size` instead
Details

See the references for original articles on the method. Computations and results may differ quite a bit, see the package vignettes for more details.

The dtm object is automatically converted to boolean.

If \texttt{min\_segment\_size} > 0 then \texttt{doc\_id} must be provided unless the corpus comes from \texttt{split\_segments}, in this case \texttt{segment\_source} is used by default.

Value

The result is a list of both class \texttt{hclust} and \texttt{rainette}. Besides the elements of an \texttt{hclust} object, two more results are available:

- \texttt{uce\_groups} give the group of each document for each \( k \)
- \texttt{group} give the group of each document for the maximum value of \( k \) available

References

- Reinert M. Une méthode de classification descendante hiérarchique : application à l’analyse lexicale par contexte, Cahiers de l’analyse des données, Volume 8, Numéro 2, 1983. \url{http://www.numdam.org/item/?id=CAD_1983__8_2_187_0}


See Also

\texttt{split\_segments()}, \texttt{rainette2()}, \texttt{cutree\_rainette()}, \texttt{rainette\_plot()}, \texttt{rainette\_explor()}

Examples

```r
require(quanteda)
corpus <- data_corpus_inaugural
corpus <- head(corpus, n = 10)
corpus <- split_segments(corpus)
tok <- tokens(corpus, remove_punct = TRUE)
tok <- tokens_remove(tok, stopwords("en"))
dtm <- dfm(tok, tolower = TRUE)
dtm <- dfm_trim(dtm, min_docfreq = 3)
res <- rainette(dtm, k = 3, min_segment_size = 15)
```
rainette2

Corpus clustering based on the Reinert method - Double clustering

Description

Corpus clustering based on the Reinert method - Double clustering

Usage

rainette2(
  x,
  y = NULL,
  max_k = 5,
  min_segment_size1 = 10,
  min_segment_size2 = 15,
  doc_id = NULL,
  min_members = 10,
  min_chi2 = 3.84,
  uc_size1,
  uc_size2,
  ...
)

Arguments

x either a quanteda dfm object or the result of rainette()
y if x is a rainette() result, this must be another rainette() result from same dfm but with different uc size.
max_k maximum number of clusters to compute
min_segment_size1 if x is a dfm, minimum uc size for first clustering
min_segment_size2 if x is a dfm, minimum uc size for second clustering
doc_id character name of a dtm docvar which identifies source documents.
min_members minimum members of each cluster
min_chi2 minimum chi2 for each cluster
uc_size1 deprecated, use min_segment_size1 instead
uc_size2 deprecated, use min_segment_size2 instead
... if x is a dfm object, parameters passed to rainette() for both simple clusterings
Details

You can pass a quanteda dfm as x object, the function then performs two simple clustering with
varying minimum uc size, and then proceed to find optimal partitions based on the results of both
clusterings.

If both clusterings have already been computed, you can pass them as x and y arguments and the
function will only look for optimal partitions.

doc_id must be provided unless the corpus comes from split_segments, in this case segment_source
is used by default.

For more details on optimal partitions search algorithm, please see package vignettes.

Value

A tibble with optimal partitions found for each available value of k as rows, and the following
columns :

- clusters list of the crossed original clusters used in the partition
- k the number of clusters
- chi2 sum of the chi2 value of each cluster
- n sum of the size of each cluster
- groups group membership of each document for this partition (NA if not assigned)

References

- Reinert M. Une méthode de classification descendante hiérarchique : application à l’analyse
  www.numdam.org/item/?id=CAD_1983__8_2_187_0
- Reinert M., Alceste une méthodologie d’analyse des données textuelles et une application:
  Aurelia De Gerard De Nerval, Bulletin de Méthodologie Sociologique, Volume 26, Numéro
  1, 1990. doi: 10.1177/075910639002600103

See Also

rainette(), cutree_rainette2(), rainette2_plot(), rainette2_explor()

Examples

require(quanteda)
corpus <- data_corpus_inaugural
corpus <- head(corpus, n = 10)
corpus <- split_segments(corpus)
tok <- tokens(corpus, remove_punct = TRUE)
tok <- tokens_remove(tok, stopwords("en"))
dtm <- dfm(tok, tolower = TRUE)
dtm <- dfm_trim(dtm, min_docfreq = 3)

res1 <- rainette(dtm, k = 5, min_segment_size = 10)
res2 <- rainette(dtm, k = 5, min_segment_size = 15)
res <- rainette2(res1, res2, max_k = 4)

rainette2_complete_groups

Complete groups membership with knn classification

Description
Starting with groups membership computed from a rainette2 clustering, every document not assigned to a cluster is reassigned using a k-nearest neighbour classification.

Usage
rainette2_complete_groups(dfm, groups, k = 1, ...)

Arguments
dfm dfm object used for rainette2 clustering.
groups group membership computed by cutree on rainette2 result.
k number of neighbours considered.
... other arguments passed to FNN::knn.

Value
Completed group membership vector.

See Also
cutree_rainette2(), FNN::knn()

rainette2_explor

Shiny gadget for rainette2 clustering exploration

Description
Shiny gadget for rainette2 clustering exploration

Usage
rainette2_explor(res, dtm = NULL, corpus_src = NULL)
Arguments

- `res`: result object of a *rainette2* clustering
- `dtm`: the dfm object used to compute the clustering
- `corpus_src`: the quanteda corpus object used to compute the dtm

Value

No return value, called for side effects.

See Also

*rainette2_plot()*

---

**Description**

Generate a clustering description plot from a *rainette2* result

**Usage**

```r
rainette2_plot(
  res,  # result object of a *rainette2* clustering
  dtm,  # the dfm object used to compute the clustering
  k = NULL,  # number of groups. If NULL, use the biggest number possible
  criterion = c("chi2", "n"),  # criterion to use to choose the best partition. chi2 means the partition with the maximum sum of chi2, n the partition with the maximum size.
  complete_groups = FALSE,  # if TRUE, documents with NA cluster are reaffected by k-means clustering initialised with current groups centers.
  type = c("bar", "cloud"),
  n_terms = 15,
  free_scales = FALSE,
  measure = c("chi2", "lr", "frequency", "docprop"),
  show_negative = FALSE,
  text_size = 10
)
```

**Arguments**

- `res`: result object of a *rainette2* clustering
- `dtm`: the dfm object used to compute the clustering
- `k`: number of groups. If NULL, use the biggest number possible
- `criterion`: criterion to use to choose the best partition. chi2 means the partition with the maximum sum of chi2, n the partition with the maximum size.
- `complete_groups`: if TRUE, documents with NA cluster are reaffected by k-means clustering initialised with current groups centers.
Shiny gadget for rainette clustering exploration

Description

Shiny gadget for rainette clustering exploration

Usage

rainette_explor(res, dtm = NULL, corpus_src = NULL)

Arguments

res result object of a rainette clustering
dtm the dfm object used to compute the clustering
corpus_src the quanteda corpus object used to compute the dtm

Value

No return value, called for side effects.

See Also

rainette_plot
Examples

```r
## Not run:
require(quanteda)
corpus <- data_corpus_inaugural
corpus <- head(corpus, n = 10)
corpus <- split_segments(corpus)
tok <- tokens(corpus, remove_punct = TRUE)
tok <- tokens_remove(tok, stopwords("en"))
dtm <- dfm(tok, tolower = TRUE)
dtm <- dfm_trim(dtm, min_docfreq = 3)
res <- rainette(dtm, k = 3, min_segment_size = 15)
rainette_explor(res, dtm, corpus)

## End(Not run)
```

**Description**

Generate a clustering description plot from a rainette result

**Usage**

```r
rainette_plot(
  res,
  dtm,
  k = NULL,
  type = c("bar", "cloud"),
  n_terms = 15,
  free_scales = FALSE,
  measure = c("chi2", "lr", "frequency", "docprop"),
  show_negative = FALSE,
  text_size = NULL
)
```

**Arguments**

- `res` result object of a rainette clustering
- `dtm` the dfm object used to compute the clustering
- `k` number of groups. If NULL, use the biggest number possible
- `type` type of term plots: barplot or wordcloud
- `n_terms` number of terms to display in keyness plots
- `free_scales` if TRUE, all the keyness plots will have the same scale
- `measure` statistics to compute
- `show_negative` if TRUE, show negative keyness features
- `text_size` font size for barplots, max word size for wordclouds
Description

Generate cluster keyness statistics from a rainette result

Usage

rainette_stats(
  groups,  
dtm,  
measure = c("chi2", "lr", "frequency", "docprop"),  
n_terms = 15,  
show_negative = TRUE,  
max_p = 0.05
)

Arguments

  groups     groups membership computed by cutree_rainette or cutree_rainette2
  dtm        the dfm object used to compute the clustering
  measure    statistics to compute
  n_terms    number of terms to display in keyness plots
  show_negative if TRUE, show negative keyness features
  max_p      maximum keyness statistic p-value
Value
A list with, for each group, a data.frame of keyness statistics for the most specific n_terms features.

See Also
quanteda.textstats::textstat_keyness(), rainette_explor(), rainette_plot()

Examples

```r
require(quanteda)
corpus <- data_corpus_inaugural
corpus <- head(corpus, n = 10)
corpus <- split_segments(corpus)
tok <- tokens(corpus, remove_punct = TRUE)
tok <- tokens_remove(tok, stopwords("en"))
dtm <- dfm(tok, tolower = TRUE)
dtm <- dfm_trim(dtm, min_docfreq = 3)
res <- rainette(dtm, k = 3, min_segment_size = 15)
groups <- cutree_rainette(res, k = 3)
rainette_stats(groups, dtm)
```

Description
Remove features from dtm of each group base don cc_test and tsj

Usage

```r
select_features(m, indices1, indices2, cc_test = 0.3, tsj = 3)
```

Arguments

- `m` global dtm
- `indices1` indices of documents of group 1
- `indices2` indices of documents of group 2
- `cc_test` maximum contingency coefficient value for the feature to be kept in both groups.
- `tsj` minimum feature frequency in the dtm

Details
Internal function, not to be used directly
Value

A list of two character vectors: cols1 is the name of features to keep in group 1, cols2 the name of features to keep in group 2.

Description

Split a character string or corpus into segments, taking into account punctuation where possible.

Usage

```r
split_segments(obj, segment_size = 40, segment_size_window = NULL)
```

## S3 method for class 'character'
`split_segments(obj, segment_size = 40, segment_size_window = NULL)`

## S3 method for class 'Corpus'
`split_segments(obj, segment_size = 40, segment_size_window = NULL)`

## S3 method for class 'corpus'
`split_segments(obj, segment_size = 40, segment_size_window = NULL)`

## S3 method for class 'tokens'
`split_segments(obj, segment_size = 40, segment_size_window = NULL)`

Arguments

- **obj**: character string, quanteda or tm corpus object
- **segment_size**: segment size (in words)
- **segment_size_window**: window around segment size to look for best splitting point

Value

If `obj` is a tm or quanteda corpus object, the result is a quanteda corpus.

Examples

```r
require(quanteda)
split_segments(data_corpus_inaugural)
```
**switch_docs**

Switch documents between two groups to maximize chi-square value

**Description**

Switch documents between two groups to maximize chi-square value

**Usage**

```r
switch_docs(m, indices, max_index, max_chisq)
```

**Arguments**

- `m` original dtm
- `indices` documents indices ordered by first CA axis coordinates
- `max_index` document index where the split is maximum
- `max_chisq` maximum chi-square value

**Details**

Internal function, not to be used directly

**Value**

A list of two vectors `indices1` and `indices2`, which contain the documents indices of each group after documents switching, and a `chisq` value, the new corresponding chi-square value after switching.
Index

cluster_tab, 3
clusters_by_doc_table, 2
clusters_by_doc_table(), 6
cutree, 4
cutree_rainette, 4
cutree_rainette(), 4, 9
cutree_rainette2, 5
cutree_rainette2(), 11, 12
docs_by_cluster_table, 5
docs_by_cluster_table(), 3

file, 6
FNN::knn(), 12

import_corpus_iramuteq, 6

merge_segments, 7

order_docs, 7

quanteda.textstats::textstat_keyness(), 14, 16, 17

rainette, 8
rainette(), 10, 11
rainette2, 10
rainette2(), 9
rainette2_complete_groups, 12
rainette2_complete_groups(), 5, 14
rainette2_explor, 12
rainette2_explor(), 11, 14
rainette2_plot, 13
rainette2_plot(), 11, 13
rainette_explor, 14
rainette_explor(), 9, 16, 17
rainette_plot, 15
rainette_plot(), 9, 17
rainette_stats, 16
rainette_stats(), 16