Package 'metasnf'

November 9, 2024

Title Meta Clustering with Similarity Network Fusion

Version 1.1.2

Description Framework to facilitate patient subtyping with similarity network fusion and meta clustering. The similarity network fusion (SNF) algorithm was introduced by Wang et al. (2014) in <doi:10.1038/nmeth.2810>. SNF is a data integration approach that can transform high-dimensional and diverse data types into a single similarity network suitable for clustering with minimal loss of information from each initial data source. The meta clustering approach was introduced by Caruana et al. (2006) in <doi:10.1109/ICDM.2006.103>. Meta clustering involves generating a wide range of cluster solutions by adjusting clustering hyperparameters, then clustering the solutions themselves into a manageable number of qualitatively similar solutions, and finally characterizing representative solutions to find ones that are best for the user's specific context. This package provides a framework to easily transform multimodal data into a wide range of similarity network fusion-derived cluster solutions as well as to visualize, characterize, and validate those solutions. Core package functionality includes easy customization of distance metrics, clustering algorithms, and SNF hyperparameters to generate diverse clustering solutions; calculation and plotting of associations between features, between patients, and between cluster solutions; and standard cluster validation approaches including resampled measures of cluster stability, standard metrics of cluster quality, and label propagation to evaluate generalizability in unseen data. Associated vignettes guide the user through using the package to identify patient subtypes while adhering to best practices for unsupervised learning.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Imports cluster, digest, dplyr, ggplot2, grDevices, MASS, mclust, methods, progressr, purrr, rlang, SNFtool, stats, tidyr, utils

Suggests circlize, ComplexHeatmap, InteractiveComplexHeatmap, clv, future, future.apply, knitr, rmarkdown, testthat (>= 3.0.0), ggalluvial, dbscan

Config/testthat/edition 3

Depends R (>= 4.1.0)

LazyData true

VignetteBuilder knitr

URL https://branchlab.github.io/metasnf/,
https://github.com/BRANCHlab/metasnf/

BugReports https://github.com/BRANCHlab/metasnf/issues

NeedsCompilation no

Author Prashanth S Velayudhan [aut, cre],
Xiaoqiao Xu [aut],
Prajkta Kallurkar [aut],
Ana Patricia Balbon [aut],
Maria T Secara [aut],
Adam Taback [aut],
Denise Sabac [aut],
Nicholas Chan [aut],
Shihao Ma [aut],
Bo Wang [aut],
Daniel Felsky [aut],
Stephanie H Ameis [aut],
Brian Cox [aut],
Colin Hawco [aut],
Lauren Erdman [aut],
Anne L Wheeler [aut, ths]

Maintainer Prashanth S Velayudhan <psvelayu@gmail.com>

Repository CRAN

Date/Publication 2024-11-08 23:20:02 UTC

Contents

`abcd_anxiety` *Mock ABCD anxiety data*

Description

A randomly shuffled and anonymized copy of anxiety data from the NIMH Data archive. The original file used was pdem02.txt. The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcdutils) function `get_cbcl_anxiety`.

Usage

```
abcd_anxiety
```

Format

`abcd_anxiety`:

A data frame with 275 rows and 2 columns:

patient The unique identifier of the ABCD dataset

cbcl_anxiety_r Ordinal value of impairment on CBCL anxiety, either 0 (no impairment), 1 (borderline clinical), or 2 (clinically impaired)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

---

abcd_colour *Mock ABCD "colour" data*

Description

A randomly shuffled and anonymized copy of depression data from the NIMH Data archive. The original file used was pdem02.txt. The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcd function `get_cbcl_depress`. The data was transformed into categorical colour values to demonstrate the Chi-squared test capabilities of `extend_solutions`.

Usage

```
abcd_colour
```

Format

`abcd_colour`:

A data frame with 275 rows and 2 columns:

patient The unique identifier of the ABCD dataset

colour Categorical transformation of `cbcl_depress`.

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

---

abcd_cort_sa *Mock ABCD cortical surface area data*

Description

A randomly shuffled and anonymized copy of cortical surface area data from the NIMH Data archive. The original file used was mrisdp10201.txt The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcdutils) function `get_cort_t`.

Usage

```
abcd_cort_sa
```

Format

`abcd_cort_sa`**:**

A data frame with 188 rows and 152 columns:

patient The unique identifier of the ABCD dataset

... Cortical surface areas of various ROIs (mm^2, I think)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and

follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

`abcd_cort_t` *Mock ABCD cortical thickness data*

Description

A randomly shuffled and anonymized copy of cortical thickness data from the NIMH Data archive. The original file used was mrisdp10201.txt The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcdutils) function `get_cort_t`.

Usage

```
abcd_cort_t
```

Format

`abcd_cort_t`**:**

A data frame with 188 rows and 152 columns:

patient The unique identifier of the ABCD dataset

... Cortical thicknesses of various ROIs (mm^3, I think)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete

listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

`abcd_depress` *Mock ABCD depression data*

Description

A randomly shuffled and anonymized copy of depression data from the NIMH Data archive. The original file used was pdem02.txt. The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcd function `get_cbcl_depress`.

Usage

```
abcd_depress
```

Format

`abcd_depress`**:**

A data frame with 275 rows and 2 columns:

patient The unique identifier of the ABCD dataset

cbcl_depress_r Ordinal value of impairment on CBCL anxiety, either 0 (no impairment), 1 (borderline clinical), or 2 (clinically impaired)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

abcd_h_income *Mock ABCD income data*

Description

Like abcd_income, but with no NAs in patient column

Usage

```
abcd_h_income
```

Format

`abcd_income`:

A data frame with 300 rows and 2 columns:

patient The unique identifier of the ABCD dataset

household_income Household income in 3 category levels (low = 1, medium = 2, high = 3)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

abcd_income *Mock ABCD income data*

Description

A randomly shuffled and anonymized copy of income data from the NIMH Data archive. The original file used was pdem02.txt The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcdutils) function `get_income`.

Usage

```
abcd_income
```

Format

`abcd_income`:
A data frame with 300 rows and 2 columns:

patient The unique identifier of the ABCD dataset

household_income Household income in 3 category levels (low = 1, medium = 2, high = 3)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

`abcd_pubertal` *Mock ABCD pubertal status data*

Description

A randomly shuffled and anonymized copy of pubertal status data from the NIMH Data archive. The original files used were abcd_ssphp01.txt and abcd_ssphy01.txt. The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcdutils) function `get_pubertal_status`.

Usage

```
abcd_pubertal
```

Format

`abcd_pubertal`:
A data frame with 275 rows and 2 columns:

patient The unique identifier of the ABCD dataset

pubertal_status Average reported pubertal status between child and parent (1-5 categorical scale)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

---

abcd_subc_v *Mock ABCD subcortical volumes data*

Description

A randomly shuffled and anonymized copy of subcortical volume data from the NIMH Data archive. The original file used was smrip10201.txt The file was pre-processed by the abcdutils package (https://github.com/BRANCHlab/abcdutils) function `get_subc_v`.

Usage

```
abcd_subc_v
```

Format

`abcd_subc_v`**:**

A data frame with 174 rows and 31 columns:

patient The unique identifier of the ABCD dataset

... Subcortical volumes of various ROIs (mm^3, I think)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and

follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

`add_columns` *Add columns to a dataframe*

Description

Add new columns to a dataframe by providing a character vector of column names (param `newcols`) and a value to occupy each row of the new columns (param `fill`, NA by default).

Usage

```
add_columns(df, newcols, fill = NA)
```

Arguments

`df`	The dataframe to extend
`newcols`	The vector containing new column names
`fill`	The values of the elements of the newly added columns. NA by default.

Value

extended_df The dataframe containing the added columns

`add_settings_matrix_rows`
Add settings matrix rows

Description

Add settings matrix rows

Usage

```
add_settings_matrix_rows(
  settings_matrix,
  seed = NULL,
  nrows = 0,
  min_removed_inputs = 0,
  max_removed_inputs = sum(startsWith(colnames(settings_matrix), "inc_")) - 1,
  dropout_dist = "exponential",
  min_alpha = NULL,
  max_alpha = NULL,
  min_k = NULL,
  max_k = NULL,
  min_t = NULL,
  max_t = NULL,
  alpha_values = NULL,
  k_values = NULL,
  t_values = NULL,
  possible_snf_schemes = c(1, 2, 3),
  clustering_algorithms = NULL,
  continuous_distances = NULL,
  discrete_distances = NULL,
  ordinal_distances = NULL,
  categorical_distances = NULL,
  mixed_distances = NULL,
  distance_metrics_list = NULL,
  snf_input_weights = NULL,
  snf_domain_weights = NULL,
  retry_limit = 10
)
```

Arguments

settings_matrix	The existing settings matrix
seed	set a seed for the random matrix generation. Setting this value will change the seed of the global environment.
nrows	Number of rows to generate for the settings matrix.
min_removed_inputs	The smallest number of input dataframes that may be randomly removed. By default, 0.
max_removed_inputs	The largest number of input dataframes that may be randomly removed. By default, this is 1 less than all the provided input dataframes in the data_list.
dropout_dist	Parameter controlling how the random removal of input dataframes should occur. Can be "none" (no input dataframes are randomly removed), "uniform" (uniformly sample between min_removed_inputs and max_removed_inputs to

determine number of input dataframes to remove), or "exponential" (pick number of input dataframes to remove by sampling from min_removed_inputs to max_removed_inputs with an exponential distribution; default).

`min_alpha` The minimum value that the alpha hyperparameter can have. Random assigned value of alpha for each row will be obtained by uniformly sampling numbers between `min_alpha` and `max_alpha` at intervals of 0.1. Cannot be used in conjunction with the `alpha_values` parameter.

`max_alpha` The maximum value that the alpha hyperparameter can have. See `min_alpha` parameter. Cannot be used in conjunction with the `alpha_values` parameter.

`min_k` The minimum value that the k hyperparameter can have. Random assigned value of k for each row will be obtained by uniformly sampling numbers between `min_k` and `max_k` at intervals of 1. Cannot be used in conjunction with the `k_values` parameter.

`max_k` The maximum value that the k hyperparameter can have. See `min_k` parameter. Cannot be used in conjunction with the `k_values` parameter.

`min_t` The minimum value that the t hyperparameter can have. Random assigned value of t for each row will be obtained by uniformly sampling numbers between `min_t` and `max_t` at intervals of 1. Cannot be used in conjunction with the `t_values` parameter.

`max_t` The maximum value that the t hyperparameter can have. See `min_t` parameter. Cannot be used in conjunction with the `t_values` parameter.

`alpha_values` A number or numeric vector of a set of possible values that alpha can take on. Value will be obtained by uniformly sampling the vector. Cannot be used in conjunction with the `min_alpha` or `max_alpha` parameters.

`k_values` A number or numeric vector of a set of possible values that k can take on. Value will be obtained by uniformly sampling the vector. Cannot be used in conjunction with the `min_k` or `max_k` parameters.

`t_values` A number or numeric vector of a set of possible values that t can take on. Value will be obtained by uniformly sampling the vector. Cannot be used in conjunction with the `min_t` or `max_t` parameters.

`possible_snf_schemes`
A vector containing the possible snf_schemes to uniformly randomly select from. By default, the vector contains all 3 possible schemes: c(1, 2, 3). 1 corresponds to the "individual" scheme, 2 corresponds to the "domain" scheme, and 3 corresponds to the "twostep" scheme.

`clustering_algorithms`
A list of clustering algorithms to uniformly randomly pick from when clustering. When not specified, randomly select between spectral clustering using the eigen-gap heuristic and spectral clustering using the rotation cost heuristic. See ?generate_clust_algs_list for more details on running custom clustering algorithms.

`continuous_distances`
A vector of continuous distance metrics to use when a custom distance_metrics_list is provided.

`discrete_distances`
: A vector of categorical distance metrics to use when a custom distance_metrics_list is provided.

`ordinal_distances`
: A vector of categorical distance metrics to use when a custom distance_metrics_list is provided.

`categorical_distances`
: A vector of categorical distance metrics to use when a custom distance_metrics_list is provided.

`mixed_distances`
: A vector of mixed distance metrics to use when a custom distance_metrics_list is provided.

`distance_metrics_list`
: List containing distance metrics to vary over. See ?generate_distance_metrics_list.

`snf_input_weights`
: Nested list containing weights for when SNF is used to merge individual input measures (see ?generate_snf_weights)

`snf_domain_weights`
: Nested list containing weights for when SNF is used to merge domains (see ?generate_snf_weights)

`retry_limit`
: The maximum number of attempts to generate a novel row. This function does not return matrices with identical rows. As the range of requested possible settings tightens and the number of requested rows increases, the risk of randomly generating a row that already exists increases. If a new random row has matched an existing row `retry_limit` number of times, the function will terminate.

Value

settings_matrix A settings matrix

`adjusted_rand_index_heatmap` *Heatmap of pairwise adjusted rand indices between solutions*

Description

Heatmap of pairwise adjusted rand indices between solutions

Usage

```
adjusted_rand_index_heatmap(
  aris,
  order = NULL,
  cluster_rows = FALSE,
  cluster_columns = FALSE,
  log_graph = FALSE,
```

```
  scale_diag = "none",
  min_colour = "#282828",
  max_colour = "firebrick2",
  col = circlize::colorRamp2(c(min(aris), max(aris)), c(min_colour, max_colour)),
  ...
)
```

Arguments

aris	Matrix of adjusted rand indices from `calc_aris()`
order	Numeric vector containing row order of the heatmap.
cluster_rows	Whether rows should be clustered.
cluster_columns	Whether columns should be clustered.
log_graph	If TRUE, log transforms the graph.
scale_diag	Method of rescaling matrix diagonals. Can be "none" (don't change diagonals), "mean" (replace diagonals with average value of off-diagonals), or "zero" (replace diagonals with 0).
min_colour	Colour used for the lowest value in the heatmap.
max_colour	Colour used for the highest value in the heatmap.
col	Colour ramp to use for the heatmap.
...	Additional parameters passed to `similarity_matrix_heatmap()`, the function that this function wraps.

Value

Returns a heatmap (class "Heatmap" from package ComplexHeatmap) that displays the pairwise adjusted Rand indices (similarities) between the cluster solutions of the provided solutions matrix.

---

age_df *Mock age data*

Description

Mock age data

Usage

```
age_df
```

Format

`age_df`**:**

A data frame with 200 rows and 2 columns:

patient_id Random three-digit number uniquely identifying the patient

age Mock age feature

Source

This data came from the SNFtool package, with slight modifications.

`alluvial_cluster_plot` *Alluvial plot of patients across cluster counts and important features*

Description

Alluvial plot of patients across cluster counts and important features

Usage

```
alluvial_cluster_plot(
  cluster_sequence,
  similarity_matrix,
  data_list = NULL,
  data = NULL,
  key_outcome,
  key_label = key_outcome,
  extra_outcomes = NULL,
  title = NULL
)
```

Arguments

`cluster_sequence`	A list of clustering algorithms (typically, the same algorithm varied over different numbers of clusters).
`similarity_matrix`	A similarity matrix.
`data_list`	A nested list of input data from `generate_data_list()`.
`data`	A dataframe that contains features to include in the plot.
`key_outcome`	The name of the feature that determines how each patient stream is coloured in the alluvial plot.
`key_label`	Name of key outcome to be used for the plot legend.
`extra_outcomes`	Names of additional features to add to the plot.
`title`	Title of the plot.

Value

An alluvial plot (class "gg" and "ggplot") showing distribution of a feature across varying number cluster solutions.

`anxiety` *Mock ABCD anxiety data*

Description

Like the mock dataframe "abcd_colour", but with "unique_id" as the "uid".

Usage

```
anxiety
```

Format

`anxiety`**:**

A data frame with 275 rows and 2 columns:

unique_id The unique identifier of the ABCD dataset

cbcl_anxiety_r Ordinal value of impairment on CBCL anxiety, either 0 (no impairment), 1 (borderline clinical), or 2 (clinically impaired)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

arrange_dl *Given a data_list object, sort data elements by subjectkey*

Description

Given a data_list object, sort data elements by subjectkey

Usage

```
arrange_dl(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

arranged_data_list The arranged data_list object

assemble_data *Collapse a dataframe and/or a data_list into a single dataframe*

Description

Collapse a dataframe and/or a data_list into a single dataframe

Usage

```
assemble_data(data, data_list)
```

Arguments

data	A dataframe.
data_list	A nested list of input data from generate_data_list().

Value

A class "data.frame" object containing all the features of the provided data frame and/or data list.

`assoc_pval_heatmap` *Heatmap of pairwise associations between features*

Description

Heatmap of pairwise associations between features

Usage

```
assoc_pval_heatmap(
  correlation_matrix,
  scale_diag = "max",
  cluster_rows = TRUE,
  cluster_columns = TRUE,
  show_row_names = TRUE,
  show_column_names = TRUE,
  show_heatmap_legend = FALSE,
  confounders = NULL,
  out_of_models = NULL,
  annotation_colours = NULL,
  labels_colour = NULL,
  split_by_domain = FALSE,
  data_list = NULL,
  significance_stars = TRUE,
  slice_font_size = 8,
  ...
)
```

Arguments

`correlation_matrix`
: Matrix containing all pairwise association p-values. The recommended way to obtain this matrix is through the calc_assoc_pval function.

`scale_diag`
: Parameter that controls how the diagonals of the correlation_matrix are adjusted in the heatmap. For best viewing, this is set to "max", which will match the diagonals to whichever pairwise association has the highest p-value.

`cluster_rows`
: Parameter for ComplexHeatmap::Heatmap. Will be ignored if split_by_domain is also provided.

`cluster_columns`
: Parameter for ComplexHeatmap::Heatmap. Will be ignored if split_by_domain is also provided.

`show_row_names`
: Parameter for ComplexHeatmap::Heatmap.

`show_column_names`
: Parameter for ComplexHeatmap::Heatmap.

`show_heatmap_legend`
: Parameter for ComplexHeatmap::Heatmap.

confounders — A named list where the elements are columns in the correlation_matrix and the names are the corresponding display names.

out_of_models — Like confounders, but a named list of out of model measures (who are also present as columns in the correlation_matrix).

annotation_colours — Named list of heatmap annotations and their colours.

labels_colour — Vector of colours to use for the columns and rows of the heatmap.

split_by_domain — The results of dl_var_summar - a dataframe that has the domain of every feature in the plotted data. columns of the correlation_matrix. Will be used to "slice" the heatmap into visually separated sections.

data_list — A nested list of input data from generate_data_list().

significance_stars — If TRUE (default), plots significance stars on heatmap cells

slice_font_size — Font size for domain separating labels.

... — Additional parameters passed into ComplexHeatmap::Heatmap.

Value

Returns a heatmap (class "Heatmap" from package ComplexHeatmap) that displays the pairwise associations between features from the provided correlation_matrix.

---

auto_plot *Automatically plot features across clusters*

Description

Given a single row of a solutions matrix and data provided through data_list and/or target_list arguments, this function will return a series of bar and/or jitter plots based on feature types.

Usage

```
auto_plot(
  solutions_matrix_row = NULL,
  data_list = NULL,
  cluster_df = NULL,
  target_list = NULL,
  return_plots = TRUE,
  save = NULL,
  jitter_width = 6,
  jitter_height = 6,
  bar_width = 6,
  bar_height = 6,
  verbose = FALSE
)
```

Arguments

solutions_matrix_row	A single row of a solutions matrix.
data_list	A data_list containing data to plot.
cluster_df	Directly provide a cluster_df rather than a solutions matrix. Useful if plotting data from label propagated results.
target_list	A target_list containing data to plot.
return_plots	If TRUE, the function will return a list of plots. If FALSE, the function will instead return the full data frame used for plotting.
save	If a string is provided, plots will be saved and this string will be used to prefix plot names.
jitter_width	Width of jitter plots if save is specified.
jitter_height	Height of jitter plots if save is specified.
bar_width	Width of bar plots if save is specified.
bar_height	Height of bar plots if save is specified.
verbose	If TRUE, print progress to console.

Value

By default, returns a list of plots (class "gg", "ggplot") with one plot for every feature in the provided data list and/or target list. If return_plots is FALSE, will instead return a single "data.frame" object containing every provided feature for every observation in long format.

---

bar_plot *Bar plot separating a feature by cluster*

Description

Bar plot separating a feature by cluster

Usage

```
bar_plot(df, feature)
```

Arguments

df	A data.frame containing cluster column and the feature to plot.
feature	The feature to plot.

Value

A bar plot (class "gg", "ggplot") showing the distribution of a feature across clusters.

batch_nmi *Calculate feature NMIs for a data_list and a derived solutions_matrix*

Description

Calculate feature NMIs for a data_list and a derived solutions_matrix

Usage

```
batch_nmi(
  data_list,
  solutions_matrix,
  clust_algs_list = NULL,
  distance_metrics_list = NULL,
  automatic_standard_normalize = FALSE,
  transpose = TRUE,
  ignore_inclusions = TRUE,
  verbose = FALSE
)
```

Arguments

data_list	A nested list of input data from generate_data_list(). Use the same value as was used in the original call to batch_snf().
solutions_matrix	Result of batch_snf storing cluster solutions and the settings that were used to generate them. Use the same value as was used in the original call to batch_snf().
clust_algs_list	List of custom clustering algorithms to apply to the final fused network. See ?generate_clust_algs_list. Use the same value as was used in the original call to batch_snf().
distance_metrics_list	An optional nested list containing which distance metric function should be used for the various feature types (continuous, discrete, ordinal, categorical, and mixed). Use the same value as was used in the original call to batch_snf().
automatic_standard_normalize	If TRUE, will automatically apply standard normalization prior to calculation of any distance matrices. Use the same value as was used in the original call to batch_snf().
transpose	If TRUE, will transpose the output dataframe.
ignore_inclusions	If TRUE, will ignore the inclusion columns in the solutions matrix and calculate NMIs for all features. If FALSE, will give NAs for features that were dropped on a given settings_matrix row.
verbose	If TRUE, print progress to console.

Value

A "data.frame" class object containing one row for every feature in the provided data list and one column for every solution in the provided solutions matrix. Populated values show the calculated NMI score for each feature-solution combination.

`batch_row_closure` *Generate closure function to run batch_snf in an apply-friendly format*

Description

Generate closure function to run batch_snf in an apply-friendly format

Usage

```
batch_row_closure(
  data_list,
  distance_metrics_list,
  clust_algs_list,
  settings_matrix,
  weights_matrix,
  similarity_matrix_dir,
  return_similarity_matrices,
  prog
)
```

Arguments

`data_list`	A nested list of input data from `generate_data_list()`.
`distance_metrics_list`	An optional nested list containing which distance metric function should be used for the various feature types (continuous, discrete, ordinal, categorical, and mixed). See ?generate_distance_metrics_list for details on how to build this.
`clust_algs_list`	List of custom clustering algorithms to apply to the final fused network. See ?generate_clust_algs_list.
`settings_matrix`	matrix indicating parameters to iterate SNF through.
`weights_matrix`	A matrix containing feature weights to use during distance matrix calculation. See ?generate_weights_matrix for details on how to build this.
`similarity_matrix_dir`	If specified, this directory will be used to save all generated similarity matrices.
`return_similarity_matrices`	If TRUE, function will return a list where the first element is the solutions matrix and the second element is a list of similarity matrices for each row in the solutions_matrix. Default FALSE.
`prog`	Progressr function to update parallel processing progress

Value

A "function" class object used to run `batch_snf` in lapply-form for parallel processing.

`batch_snf` *Run variations of SNF.*

Description

This is the core function of the metasnf package. Using the information stored in a settings_matrix (see ?generate_settings_matrix) and a data_list (see ?generate_data_list), run repeated complete SNF pipelines to generate a broad space of post-SNF cluster solutions.

Usage

```
batch_snf(
  data_list,
  settings_matrix,
  processes = 1,
  return_similarity_matrices = FALSE,
  similarity_matrix_dir = NULL,
  clust_algs_list = NULL,
  suppress_clustering = FALSE,
  distance_metrics_list = NULL,
  weights_matrix = NULL,
  automatic_standard_normalize = FALSE,
  verbose = FALSE
)
```

Arguments

`data_list` A nested list of input data from `generate_data_list()`.

`settings_matrix`
A data.frame where each row completely defines an SNF pipeline transforming individual input dataframes into a final cluster solution. See ?generate_settings_matrix or https://branchlab.github.io/metasnf/articles/settings_matrix.html for more details.

`processes` Specify number of processes used to complete SNF iterations

- `1` (default) Sequential processing: function will iterate through the `settings_matrix` one row at a time with a for loop. This option will not make use of multiple CPU cores, but will show a progress bar.
- `2` or higher: Parallel processing will use the `future.apply::future_apply` to distribute the SNF iterations across the specified number of CPU cores. If higher than the number of available cores, a warning will be printed and the maximum number of cores will be used.
- `max`: All available cores will be used.

`return_similarity_matrices`
: If TRUE, function will return a list where the first element is the solutions matrix and the second element is a list of similarity matrices for each row in the solutions_matrix. Default FALSE.

`similarity_matrix_dir`
: If specified, this directory will be used to save all generated similarity matrices.

`clust_algs_list`
: List of custom clustering algorithms to apply to the final fused network. See ?generate_clust_algs_list.

`suppress_clustering`
: If FALSE (default), will apply default or custom clustering algorithms to provide cluster solutions on every iteration of SNF. If TRUE, parameter `similarity_matrix_dir` must be specified.

`distance_metrics_list`
: An optional nested list containing which distance metric function should be used for the various feature types (continuous, discrete, ordinal, categorical, and mixed). See ?generate_distance_metrics_list for details on how to build this.

`weights_matrix`
: A matrix containing feature weights to use during distance matrix calculation. See ?generate_weights_matrix for details on how to build this.

`automatic_standard_normalize`
: If TRUE, will automatically apply standard normalization prior to calculation of any distance matrices. This parameter cannot be used in conjunction with a custom distance metrics list. If you wish to supply custom distance metrics but also always have standard normalization, simply ensure that the numeric (continuous, discrete, and ordinal) distance metrics are only populated with distance metric functions that apply standard normalization. See https://branchlab.github.io/metasnf/articles/distance_
to learn more.

`verbose`
: If TRUE, print time remaining estimates to console.

Value

By default, returns a solutions matrix (class "data.frame"), a a data frame containing one row for every row of the provided settings matrix, all the original columns of that settings matrix, and new columns containing the assigned cluster of each observation from the cluster solution derived by that row's settings. If `return_similarity_matrices` is TRUE, the function will instead return a list containing the solutions matrix as well as a list of the final similarity matrices (class "matrix") generated by SNF for each row of the settings matrix. If `suppress_clustering` is TRUE, the solutions matrix will not be returned in the output.

`batch_snf_subsamples` *Run SNF clustering pipeline on a list of subsampled data lists.*

Description

Run SNF clustering pipeline on a list of subsampled data lists.

Usage

```
batch_snf_subsamples(
  data_list_subsamples,
  settings_matrix,
  processes = 1,
  return_similarity_matrices = FALSE,
  clust_algs_list = NULL,
  suppress_clustering = FALSE,
  distance_metrics_list = NULL,
  weights_matrix = NULL,
  automatic_standard_normalize = FALSE,
  return_solutions_matrices = FALSE,
  verbose = FALSE
)
```

Arguments

`data_list_subsamples`
: A list of subsampled data lists. This object is generated by the function `batch_snf_subsamples()`.

`settings_matrix`
: A settings matrix defining the parameters of the SNF pipelines to be applied to the subsampled data lists.

`processes`
: See ?batch_snf.

`return_similarity_matrices`
: See ?batch_snf.

`clust_algs_list`
: See ?batch_snf.

`suppress_clustering`
: See ?batch_snf.

`distance_metrics_list`
: See ?batch_snf.

`weights_matrix`
: See ?batch_snf.

`automatic_standard_normalize`
: See ?batch_snf.

`return_solutions_matrices`
: If TRUE, includes the solutions matrices corresponding to each subsample in the output.

`verbose`
: If TRUE, print time remaining estimates to console.

Value

By default, returns a one-element list: `cluster_solutions`, which is itself a list of cluster solution data frames corresponding to each of the provided data list subsamples. Setting the parameters `return_similarity_matrices` and `return_solutions_matrices` to `TRUE` will turn the result of the function to a three-element list containing the corresponding solutions matrices and final fused similarity matrices of those cluster solutions, should you require these objects for your own stability calculations.

`calculate_coclustering` *Calculate coclustering data.*

Description

Calculate coclustering data.

Usage

```
calculate_coclustering(subsample_solutions, solutions_matrix, verbose = FALSE)
```

Arguments

`subsample_solutions`	A list of containing cluster solutions from distinct subsamples of the data. This object is generated by the function `batch_snf_subsamples()`. These solutions should correspond to the ones in the solutions matrix.
`solutions_matrix`	A solutions matrix. This object is generated by the function `batch_snf()`. The solutions in the solutions matrix should correspond to those in the subsample solutions.
`verbose`	If TRUE, print time remaining estimates to console.

Value

A list containing the following components:

- cocluster_dfs: A list of dataframes, one per cluster solution, that shows the number of times that every pair of subjects in the original cluster solution occurred in the same subsample, the number of times that every pair clustered together in a subsample, and the corresponding fraction of times that every pair clustered together in a subsample.
- cocluster_ss_mats: The number of times every pair of subjects occurred in the same subsample, formatted as a pairwise matrix.
- cocluster_sc_mats: The number of times every pair of subjects occurred in the same cluster, formatted as a pairwise matrix.
- cocluster_cf_mats: The fraction of times every pair of subjects occurred in the same cluster, formatted as a pairwise matrix.
- cocluster_summary: Specifically among pairs of subjects that clustered together in the original full cluster solution, what fraction of those pairs remained clustered together throughout the subsample solutions. This information is formatted as a dataframe with one row per cluster solution.

calculate_db_indices *Calculate Davies-Bouldin indices*

Description

Given a solutions_matrix and a list of similarity_matrices (or a single similarity_matrix if the solutions_matrix has only 1 row), return a vector of Davies-Bouldin indices

Usage

```
calculate_db_indices(solutions_matrix, similarity_matrices)
```

Arguments

solutions_matrix	A solutions_matrix (see ?batch_snf)
similarity_matrices	A list of similarity matrices (see ?batch_snf)

Value

davies_bouldin_indices A vector of Davies-Bouldin indices for each cluster solution.

calculate_dunn_indices *Calculate Dunn indices*

Description

Given a solutions_matrix and a list of similarity_matrices (or a single similarity_matrix if the solutions_matrix has only 1 row), return vector of Dunn indices

Usage

```
calculate_dunn_indices(solutions_matrix, similarity_matrices)
```

Arguments

solutions_matrix	A solutions_matrix (see ?batch_snf)
similarity_matrices	A list of similarity matrices (see ?batch_snf)

Value

dunn_indices A vector of Dunn indices for each cluster solution

calculate_silhouettes *Calculate silhouette scores*

Description

Given a solutions_matrix and a list of similarity_matrices (or a single similarity_matrix if the solutions_matrix has only 1 row), return a list of 'silhouette' objects from the cluster package

Usage

```
calculate_silhouettes(solutions_matrix, similarity_matrices)
```

Arguments

solutions_matrix	A solutions_matrix (see ?batch_snf)
similarity_matrices	A list of similarity matrices (see ?batch_snf)

Value

silhouette_scores A list of "silhouette" objects from the cluster package.

calc_aris *Meta-cluster calculations*

Description

Generate matrix of pairwise cluster-solution similarities by Adjusted Rand index calculations.

Usage

```
calc_aris(solutions_matrix, processes = 1, verbose = FALSE)
```

Arguments

solutions_matrix	solutions_matrix containing cluster solutions to calculate pairwise ARIs for.
processes	Specify number of processes used to complete calculations • 1 (default) Sequential processing • 2 or higher: Parallel processing will use the future.apply::future_apply to distribute the calculations across the specified number of CPU cores. If higher than the number of available cores, a warning will be printed and the maximum number of cores will be used. • max: All available cores will be used. Note that no progress indicator is available during multi-core processing.
verbose	If TRUE, print progress to console.

Value

om_aris ARIs between clustering solutions of an solutions matrix

---

`calc_assoc_pval` *Calculate p-values based on feature vectors and their types*

Description

Calculate p-values based on feature vectors and their types

Usage

```
calc_assoc_pval(var1, var2, type1, type2, cat_test = "chi_squared")
```

Arguments

`var1`	A single vector containing a feature.
`var2`	A single vector containing a feature.
`type1`	The type of var1 (continuous, discrete, ordinal, categorical).
`type2`	The type of var2 (continuous, discrete, ordinal, categorical).
`cat_test`	String indicating which statistical test will be used to associate cluster with a categorical feature. Options are "chi_squared" for the Chi-squared test and "fisher_exact" for Fisher's exact test.

Value

pval A p-value from a statistical test based on the provided types. Currently, this will either be the F-test p-value from a linear model if at least one feature is non-categorical, or the chi-squared test p-value if both features are categorical.

`calc_assoc_pval_matrix`
Calculate p-values for all pairwise associations of features in a data_list

---

Description

Calculate p-values for all pairwise associations of features in a data_list

Usage

```
calc_assoc_pval_matrix(data_list, verbose = FALSE, cat_test = "chi_squared")
```

Arguments

data_list	A nested list of input data from generate_data_list().
verbose	If TRUE, prints new line everytime a p-value is being calculated.
cat_test	String indicating which statistical test will be used to associate cluster with a categorical feature. Options are "chi_squared" for the Chi-squared test and "fisher_exact" for Fisher's exact test.

Value

A "matrix" class object containing pairwise association p-values between the features in the provided data list.

---

cancer_diagnosis_df *Mock diagnosis data*

Description

This is the same data as diagnosis_df, with renamed features and columns.

Usage

```
cancer_diagnosis_df
```

Format

cancer_diagnosis_df:

A data frame with 200 rows and 2 columns:

patient_id Random three-digit number uniquely identifying the patient

diagnosis Mock cancer diagnosis feature (1, 2, or 3)

Source

This data came from the SNFtool package, with slight modifications.

cell_significance_fn *Place significance stars on ComplexHeatmap cells.*

Description

This is an internal function meant to be used to by the assoc_pval_heatmap function.

Usage

```
cell_significance_fn(data)
```

Arguments

data	The matrix containing the cells to base the significance stars on.

Value

cell_fn Another function that is well-formatted for usage as the cell_fun argument in ComplexHeatmap::Heatmap.

char_to_fac *Convert character-type columns of a dataframe to factor-type*

Description

Convert character-type columns of a dataframe to factor-type

Usage

```
char_to_fac(df)
```

Arguments

df	A dataframe

Value

df_converted The dataframe with factor-type columns instead of char-type columns

check_dataless_annotations *Helper function to stop annotation building when no data was provided*

Description

Helper function to stop annotation building when no data was provided

Usage

```
check_dataless_annotations(annotation_requests, data)
```

Arguments

annotation_requests	A list of requested annotations
data	A dataframe with data to build annotations

Value

Does not return any value. This function just raises an error when annotations are requested without any provided data for a heatmap.

check_hm_dependencies *Check for ComplexHeatmap and circlize dependencies*

Description

Check for ComplexHeatmap and circlize dependencies

Usage

```
check_hm_dependencies()
```

Value

Does not return any value. This function just checks that the ComplexHeatmap and circlize packages are installed.

check_similarity_matrices *Check validity of similarity matrices*

Description

Check to see if similarity matrices in a list have the following properties:

1. The maximum value in the entire matrix is 0.5
2. Every value in the diagonal is 0.5

Usage

```
check_similarity_matrices(similarity_matrices)
```

Arguments

similarity_matrices	A list of similarity matrices

Value

valid_matrices Boolean indicating if properties are met by all similarity matrices

chi_squared_pval *Chi-squared test p-value (generic)*

Description

Return p-value for chi-squared test for any two features

Usage

```
chi_squared_pval(cat_var1, cat_var2)
```

Arguments

cat_var1	A categorical feature.
cat_var2	A categorical feature.

Value

pval A p-value (class "numeric").

coclustering_coverage_check *Coclustering coverage check*

Description

Check if coclustered data has at least one subsample in which every pair of subjects were a part of simultaneously.

Usage

```
coclustering_coverage_check(cocluster_df, action = "warn")
```

Arguments

cocluster_df	Dataframe containing coclustering data.
action	Control if parent function should warn or stop.

Value

This function does not return any value. It checks a `cocluster_df` for complete coverage (all pairs occur in the same solution at least once). Will raise a warning or error if coverage is incomplete depending on the value of the action parameter.

cocluster_density *Density plot coclustering stability across subsampled data.*

Description

This function creates a density plot that shows, for all pairs of observations that originally clustered together, the distribution of the the fractions that those pairs clustered together across subsampled data.

Usage

```
cocluster_density(cocluster_df)
```

Arguments

cocluster_df	A dataframe containing coclustering data for a single cluster solution. This object is generated by the `calculate_coclustering` function.

Value

Density plot (class "gg", "ggplot") of the distribution of coclustering across pairs and subsamples of the data.

cocluster_heatmap *Heatmap of observation co-clustering across resampled data.*

Description

Create a heatmap that shows the distribution of observation co-clustering across resampled data.

Usage

```
cocluster_heatmap(
  cocluster_df,
  cluster_rows = TRUE,
  cluster_columns = TRUE,
  show_row_names = FALSE,
  show_column_names = FALSE,
  data_list = NULL,
  data = NULL,
  left_bar = NULL,
  right_bar = NULL,
  top_bar = NULL,
  bottom_bar = NULL,
  left_hm = NULL,
  right_hm = NULL,
  top_hm = NULL,
  bottom_hm = NULL,
  annotation_colours = NULL,
  min_colour = NULL,
  max_colour = NULL,
  ...
)
```

Arguments

cocluster_df	A dataframe containing coclustering data for a single cluster solution. This object is generated by the calculate_coclustering function.
cluster_rows	Argument passed to ComplexHeatmap::Heatmap().
cluster_columns	Argument passed to ComplexHeatmap::Heatmap().
show_row_names	Argument passed to ComplexHeatmap::Heatmap().
show_column_names	Argument passed to ComplexHeatmap::Heatmap().
data_list	See ?similarity_matrix_heatmap.
data	See ?similarity_matrix_heatmap.
left_bar	See ?similarity_matrix_heatmap.
right_bar	See ?similarity_matrix_heatmap.

top_bar	See ?similarity_matrix_heatmap.
bottom_bar	See ?similarity_matrix_heatmap.
left_hm	See ?similarity_matrix_heatmap.
right_hm	See ?similarity_matrix_heatmap.
top_hm	See ?similarity_matrix_heatmap.
bottom_hm	See ?similarity_matrix_heatmap.
annotation_colours	See ?similarity_matrix_heatmap.
min_colour	See ?similarity_matrix_heatmap.
max_colour	See ?similarity_matrix_heatmap.
...	Arguments passed to ComplexHeatmap::Heatmap().

Value

Heatmap (class "Heatmap" from ComplexHeatmap) object showing the distribution of observation co-clustering across resampled data.

---

collapse_dl *Collapse a data_list into a single dataframe*

Description

Collapse a data_list into a single dataframe

Usage

```
collapse_dl(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

A "data.frame"-formatted version of the provided data list.

colour_scale *Return a colour ramp for a given vector*

Description

Given a numeric vector and min and max colour values, return a colour ramp that assigns a colour to each element in the vector. This function is a wrapper for `circlize::colorRamp2`.'

Usage

```
colour_scale(data, min_colour, max_colour)
```

Arguments

`data`	Vector of numeric values.
`min_colour`	Minimum colour value.
`max_colour`	Maximum colour value.

Value

A "function" class object that can build a circlize-style colour ramp.

convert_uids *Convert unique identifiers of data_list to 'subjectkey'*

Description

Column name "subjectkey" is reserved for the unique identifier of subjects. This function ensures all dataframes have their UID set as "subjectkey".

Usage

```
convert_uids(data_list, uid = NULL)
```

Arguments

`data_list`	A nested list of input data from `generate_data_list()`.
`uid`	(string) the name of the uid column currently used data

Value

dl_renamed_id data list with 'subjectkey' as UID

cort_sa *Mock ABCD cortical surface area data*

Description

Like the mock dataframe "abcd_cort_sa", but with "unique_id" as the "uid".

Usage

```
cort_sa
```

Format

`cort_sa`**:**

A data frame with 188 rows and 152 columns:

unique_id The unique identifier of the ABCD dataset

... Cortical surface areas of various ROIs (mm^2, I think)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

cort_t *Mock ABCD cortical thickness data*

Description

Like the mock dataframe "abcd_cort_t", but with "unique_id" as the "uid".

Usage

```
cort_t
```

Format

`cort_t`:
A data frame with 188 rows and 152 columns:

unique_id The unique identifier of the ABCD dataset

... Cortical thicknesses of various ROIs (mm^3, I think)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

---

depress *Mock ABCD depression data*

Description

Like the mock dataframe "abcd_depress", but with "unique_id" as the "uid".

Usage

```
depress
```

Format

`depress`:
A data frame with 275 rows and 2 columns:

unique_id The unique identifier of the ABCD dataset

cbcl_depress_r Ordinal value of impairment on CBCL anxiety, either 0 (no impairment), 1 (borderline clinical), or 2 (clinically impaired)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

`diagnosis_df` *Mock diagnosis data*

Description

This is the same data as `cancer_diagnosis_df`, with renamed features and columns.

Usage

```
diagnosis_df
```

Format

`diagnosis_df`**:**

A data frame with 200 rows and 2 columns:

patient_id Random three-digit number uniquely identifying the patient

diagnosis Mock diagnosis feature

Source

This data came from the SNFtool package, with slight modifications.

discretisation *Internal function for* estimate_nclust_given_graph

Description

Internal function taken from SNFtool to use for number of cluster estimation.

Usage

```
discretisation(eigenvectors)
```

Arguments

eigenvectors	Matrix of eigenvectors.

Value

"Matrix" class object, intermediate product in spectral clustering.

discretisation_evec_data *Internal function for* estimate_nclust_given_graph

Description

Internal function taken from SNFtool to use for number of cluster estimation.

Usage

```
discretisation_evec_data(eigenvector)
```

Arguments

eigenvector	Matrix of eigenvectors

Value

"Matrix" class object discretizing provided eigenvector to values 0 or 1.

dl_has_duplicates *Check if data list contains any duplicate features*

Description

Check if data list contains any duplicate features

Usage

```
dl_has_duplicates(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

Doesn't return any value. Raises warning if there are features with duplicate names in a generated data list.

dl_uid_first_col *Make the subjectkey UID columns of a data_list first*

Description

Make the subjectkey UID columns of a data_list first

Usage

```
dl_uid_first_col(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

A data list ("list"-class object) in which each data-subcomponent has "subjectkey" positioned as its first column.

dl_variable_summary *Variable-level summary of a data_list*

Description

Variable-level summary of a data_list

Usage

```
dl_variable_summary(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

variable_level_summary A dataframe containing the name, type, and domain of every variable in a data_list.

domains *Domains*

Description

Domains

Usage

```
domains(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

domain_list list of domains

`domain_merge` *SNF scheme: Domain merge*

Description

Given a data_list, returns a new data_list where all data objects of a particlar domain have been concatenated.

Usage

```
domain_merge(
  data_list,
  cont_dist_fn,
  disc_dist_fn,
  ord_dist_fn,
  cat_dist_fn,
  mix_dist_fn,
  weights_row,
  k,
  alpha,
  t
)
```

Arguments

`data_list`	A nested list of input data from `generate_data_list()`.
`cont_dist_fn`	distance metric function for continuous data.
`disc_dist_fn`	distance metric function for discrete data.
`ord_dist_fn`	distance metric function for ordinal data.
`cat_dist_fn`	distance metric function for categorical data.
`mix_dist_fn`	distance metric function for mixed data.
`weights_row`	dataframe row containing feature weights.
`k`	k hyperparameter.
`alpha`	alpha/eta/sigma hyperparameter.
`t`	SNF number of iterations hyperparameter.

Value

fused_network The final fused network (class "matrix", "array") generated by SNF.

drop_inputs *Execute inclusion*

Description

Given a data list and a settings matrix row, returns a data list of selected inputs

Usage

```
drop_inputs(settings_matrix_row, data_list)
```

Arguments

settings_matrix_row	Row of a settings matrix.
data_list	A nested list of input data from generate_data_list().

Value

A data list (class "list") in which any component with a corresponding 0 value in the provided settings matrix row has been removed.

esm_manhattan_plot *Manhattan plot of feature-cluster association p-values*

Description

Manhattan plot of feature-cluster association p-values

Usage

```
esm_manhattan_plot(
  esm,
  neg_log_pval_thresh = 5,
  threshold = NULL,
  point_size = 5,
  jitter_width = 0.1,
  jitter_height = 0.1,
  text_size = 15,
  plot_title = NULL,
  hide_x_labels = FALSE,
  bonferroni_line = FALSE
)
```

Arguments

esm	Extended solutions matrix storing associations between features and cluster assignments. See ?extend_solutions.
neg_log_pval_thresh	Threshold for negative log p-values.
threshold	P-value threshold to plot dashed line at.
point_size	Size of points in the plot.
jitter_width	Width of jitter.
jitter_height	Height of jitter.
text_size	Size of text in the plot.
plot_title	Title of the plot.
hide_x_labels	If TRUE, hides x-axis labels.
bonferroni_line	If TRUE, plots a dashed black line at the Bonferroni-corrected equivalent of the p-value threshold.

Value

A Manhattan plot (class "gg", "ggplot") showing the association p-values of features against each solution in the provided solutions matrix.

---

estimate_nclust_given_graph

Estimate number of clusters for a similarity matrix

Description

Calculate eigengap and rotation-cost estimates of the number of clusters to use when clustering a similarity matrix. This function was adapted from SNFtool::estimateClustersGivenGraph, but scales up the Laplacian operator prior to eigenvalue calculations to minimize the risk of floating point-related errors.

Usage

```
estimate_nclust_given_graph(W, NUMC = 2:10)
```

Arguments

W	Similarity matrix to calculate number of clusters for.
NUMC	Range of cluster counts to consider among when picking best number of clusters.

Value

A list containing the top two eigengap and rotation-cost estimates for the number of clusters in a given similarity matrix.

euclidean_distance *Distance metric: Euclidean distance*

Description

Distance metric: Euclidean distance

Usage

```
euclidean_distance(df, weights_row)
```

Arguments

df	Dataframe containing at least 1 data column
weights_row	Single-row dataframe where the column names contain the column names in df and the row contains the corresponding weights_row.

Value

distance_matrix A distance matrix.

expression_df *Modification of SNFtool mock dataframe "Data1"*

Description

Modification of SNFtool mock dataframe "Data1"

Usage

```
expression_df
```

Format

`expression_df`**:**
A data frame with 200 rows and 3 columns:

gene_1_expression Mock gene expression feature

gene_2_expression Mock gene expression feature

patient_id Random three-digit number uniquely identifying the patient

Source

This data came from the SNFtool package, with slight modifications.

`extend_solutions` *Extend an solutions matrix to include outcome evaluations*

Description

Extend an solutions matrix to include outcome evaluations

Usage

```
extend_solutions(
  solutions_matrix,
  target_list = NULL,
  data_list = NULL,
  cat_test = "chi_squared",
  calculate_summaries = TRUE,
  min_pval = 1e-10,
  processes = 1,
  verbose = FALSE
)
```

Arguments

`solutions_matrix`	Result of `batch_snf` storing cluster solutions and the settings that were used to generate them.
`target_list`	A data_list with features to calculate p-values for. Features in the target list will be included during p-value summary measure calculations.
`data_list`	A data_list with features to calcualte p-values for, but that should not be incorporated into p-value summary measure columns (i.e., min/mean/max p-value columns).
`cat_test`	String indicating which statistical test will be used to associate cluster with a categorical feature. Options are "chi_squared" for the Chi-squared test and "fisher_exact" for Fisher's exact test.
`calculate_summaries`	If TRUE, the function will calculate the minimum and mean p-values for each row of the solutions matrix.
`min_pval`	If assigned a value, any p-value less than this will be replaced with this value.
`processes`	The number of processes to use for parallelization. Progress is only reported for sequential processing (processes = 1).
`verbose`	If TRUE, print progress to console.

Value

extended_solutions_matrix an extended solutions matrix that contains p-value columns for each outcome in the provided target_list

fav_colour *Mock ABCD "colour" data*

Description

Like the mock dataframe "abcd_colour", but with "unique_id" as the "uid".

Usage

```
fav_colour
```

Format

`fav_colour`**:**

A data frame with 275 rows and 2 columns:

unique_id The unique identifier of the ABCD dataset

colour Categorical transformation of `cbcl_depress`.

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

fisher_exact_pval *Fisher exact test p-value*

Description

Return p-value for Fisher exact test for any two features

Usage

```
fisher_exact_pval(cat_var1, cat_var2)
```

Arguments

cat_var1	A categorical feature.
cat_var2	A categorical feature.

Value

pval A p-value (class "numeric").

---

gender_df *Mock gender data*

Description

Mock gender data

Usage

```
gender_df
```

Format

gender_df:

A data frame with 200 rows and 2 columns:

patient_id Random three-digit number uniquely identifying the patient

gender_df Mock gene methylation feature

Source

This data came from the SNFtool package, with slight modifications.

`generate_annotations_list` *Generate annotations list*

Description

Intermediate function that takes in formatted lists of features and the annotations they should be viewed through and returns annotation objects usable by ComplexHeatmap::Heatmap.

Usage

```
generate_annotations_list(
  df,
  left_bar = NULL,
  right_bar = NULL,
  top_bar = NULL,
  bottom_bar = NULL,
  left_hm = NULL,
  right_hm = NULL,
  top_hm = NULL,
  bottom_hm = NULL,
  show_legend = TRUE,
  annotation_colours = NULL
)
```

Arguments

`df`	Dataframe containing all the data that is specified in the remaining arguments.
`left_bar`	Named list of strings, where the strings are features in df that should be used for a barplot annotation on the left of the plot and the names are the names that will be used to caption the plots and their legends.
`right_bar`	See left_bar.
`top_bar`	See left_bar.
`bottom_bar`	See left_bar.
`left_hm`	Like left_bar, but with a heatmap annotation instead of a barplot annotation.
`right_hm`	See left_hm.
`top_hm`	See left_hm.
`bottom_hm`	See left_hm.
`show_legend`	Add legends to the annotations.
`annotation_colours`	Named list of heatmap annotations and their colours.

Value

annotations_list A named list of all the annotations.

`generate_clust_algs_list` *Generate a list of custom clustering algorithms*

Description

This function can be used to specify custom clustering algorithms to apply to the final similarity matrices produced by each run of the batch_snf function.

Usage

```
generate_clust_algs_list(..., disable_base = FALSE)
```

Arguments

`...`	An arbitrary number of named clustering functions (see examples below)
`disable_base`	If TRUE, do not prepend the base clustering algorithms (spectral_eigen and spectral_rot, which apply spectral clustering and use the eigen-gap and rotation cost heuristics respectively for determining the number of clusters in the graph.

Value

A list of clustering algorithm functions that can be passed into the batch_snf and generate_settings_list functions.

Examples

```
# Using just the base clustering algorithms --------------------------------
# This will just contain spectral_eigen and spectral_rot
clust_algs_list <- generate_clust_algs_list()

# Adding algorithms provided by the package --------------------------------
# This will contain the base clustering algorithms (spectral_eigen,
#  spectral_rot) as well as two pre-defined spectral clustering functions
#  that force the number of clusters to be two or five
clust_algs_list <- generate_clust_algs_list(
    "two_cluster_spectral" = spectral_two,
    "five_cluster_spectral" = spectral_five
)

# Adding your own algorithms -----------------------------------------------
# This will contain the base and user-provided clustering algorithms
my_clustering_algorithm <- function(similarity_matrix) {
    # your code that converts similarity matrix to clusters here...
    # solution_data <- list(
    #     "solution" = solution,
    #     "nclust" = number_of_clusters
    # )
```

```
    # return(solution_data)
}

# Suppress the base algorithms-------------------------------------------------
# This will contain only user-provided clustering algorithms

clust_algs_list <- generate_clust_algs_list(
    "two_cluster_spectral" = spectral_two,
    "five_cluster_spectral" = spectral_five,
    disable_base = TRUE
)
```

`generate_data_list` *Generate a data_list*

Description

This function generates the major data object that will be processed when iterating through the each SNF pipeline defined in the settings_matrix. The data_list is a named and nested list containing input dataframes (data), the name of that input dataframe (for the user's reference), the 'domain' of that dataframe (the broader source of information that the input dataframe is capturing, determined by user's domain knowledge), and the type of feature stored in the dataframe (continuous, discrete, ordinal, categorical, or mixed).

Usage

```
generate_data_list(
  ...,
  uid = NULL,
  test_subjects = NULL,
  train_subjects = NULL,
  sort_subjects = TRUE,
  remove_missing = TRUE,
  return_missing = FALSE
)
```

Arguments

`...`	Any number of list formatted as (df, "df_name", "df_domain", "df_type") OR any number of lists of lists formatted as (df, "df_name", "df_domain", "df_type")
`uid`	(string) the name of the uid column currently used data
`test_subjects`	character vector of test subjects (useful if building a full data list for label propagation)
`train_subjects`	character vector of train subjects (useful if building a full data list for label propagation)
`sort_subjects`	If TRUE, the subjects in the data_list will be sorted

`remove_missing` If TRUE (default), subjects with incomplete data will be dropped from data_list creation. Setting this value to FALSE may lead to unusual and/or unstable results during SNF, clustering, p-value calculations or label propagation.

`return_missing` If TRUE, function returns a list where the first element is the data_list and the second element is a vector of unique IDs of patients who were removed during the complete data filtration step.

Value

A nested "list" class object. Each list component contains a 4-item list of a data frame, the user-assigned name of the data frame, the user-assigned domain of the data frame, and the user-labeled type of the data frame.

Examples

```
heart_rate_df <- data.frame(
    patient_id = c("1", "2", "3"),
    var1 = c(0.04, 0.1, 0.3),
    var2 = c(30, 2, 0.3)
)

personality_test_df <- data.frame(
    patient_id = c("1", "2", "3"),
    var3 = c(900, 1990, 373),
    var4 = c(509, 2209, 83)
)

survey_response_df <- data.frame(
    patient_id = c("1", "2", "3"),
    var5 = c(1, 3, 3),
    var6 = c(2, 3, 3)
)

city_df <- data.frame(
    patient_id = c("1", "2", "3"),
    var7 = c("toronto", "montreal", "vancouver")
)

# Explicitly (Name each nested list element):
data_list <- generate_data_list(
    list(
        data = heart_rate_df,
        name = "heart_rate",
        domain = "clinical",
        type = "continuous"
    ),
    list(
        data = personality_test_df,
        name = "personality_test",
        domain = "surveys",
        type = "continuous"
    ),
```

```
    list(
        data = survey_response_df,
        name = "survey_response",
        domain = "surveys",
        type = "ordinal"
    ),
    list(
        data = city_df,
        name = "city",
        domain = "location",
        type = "categorical"
    ),
    uid = "patient_id"
)

# Compact loading
data_list <- generate_data_list(
    list(heart_rate_df, "heart_rate", "clinical", "continuous"),
    list(personality_test_df, "personality_test", "surveys", "continuous"),
    list(survey_response_df, "survey_response", "surveys", "ordinal"),
    list(city_df, "city", "location", "categorical"),
    uid = "patient_id"
)

# Printing data_list summaries
summarize_dl(data_list)

# Alternative loading: providing a single list of lists
list_of_lists <- list(
    list(heart_rate_df, "data1", "domain1", "continuous"),
    list(personality_test_df, "data2", "domain2", "continuous")
)

dl <- generate_data_list(
    list_of_lists,
    uid = "patient_id"
)
```

`generate_distance_metrics_list` *Generate a list of distance metrics*

Description

This function can be used to specify custom distance metrics

Usage

```
generate_distance_metrics_list(
  continuous_distances = NULL,
```

```
  discrete_distances = NULL,
  ordinal_distances = NULL,
  categorical_distances = NULL,
  mixed_distances = NULL,
  keep_defaults = TRUE
)
```

Arguments

continuous_distances	A named list of distance metric functions
discrete_distances	A named list of distance metric functions
ordinal_distances	A named list of distance metric functions
categorical_distances	A named list of distance metric functions
mixed_distances	A named list of distance metric functions
keep_defaults	If TRUE (default), prepend the base distance metrics (euclidean and standard normalized euclidean)

Value

distance_metrics_list A well-formatted list of distance metrics

Examples

```
# Using just the base distance metrics  ------------------------------------
distance_metrics_list <- generate_distance_metrics_list()

# Adding your own metrics ---------------------------------------------------
# This will contain the base and user-provided clustering algorithms
my_distance_metric <- function(df) {
    # your code that converts a dataframe to a distance metric here...
    # return(distance_metric)
}

distance_metrics_list <- generate_distance_metrics_list(
    continuous_distances = list(
         "my_distance_metric" = my_distance_metric
    )
)

# Suppress the base metrics-------------------------------------------------
# This will contain only user-provided clustering algorithms

distance_metrics_list <- generate_distance_metrics_list(
    continuous_distances = list(
         "my_distance_metric" = my_distance_metric
    ),
```

```
    discrete_distances = list(
        "my_distance_metric" = my_distance_metric
    ),
    ordinal_distances = list(
        "my_distance_metric" = my_distance_metric
    ),
    categorical_distances = list(
        "my_distance_metric" = my_distance_metric
    ),
    mixed_distances = list(
        "my_distance_metric" = my_distance_metric
    ),
    keep_defaults = FALSE
)
```

`generate_settings_matrix` *Build a settings matrix*

Description

The settings_matrix is a dataframe whose rows completely specify the hyperparameters and decisions required to transform individual input dataframes (found in a data_list, see ?generate_data_list) into a single similarity matrix through SNF. The format of the settings matrix is as follows:

- A column named "row_id": This column is used to keep track of the rows and should have integer values only.
- A column named "alpha": This column contains the value of the alpha hyperparameter that will be used on that run of the SNF pipeline.
- A column named "k": Like above, but for the K (nearest neighbours) hyperparameter.
- A column named "t": Like above, but for the t (number of iterations) hyperparameter.
- A column named "clust_alg": Specification of which clustering algorithm will be applied to the final similarity matrix to identify patient subtypes. By default, this column can take on the integer values 1 or 2, which correspond to spectral clustering where the number of clusters is determined by the eigen-gap or rotation cost heuristic respectively. You can learn more about this parameter here: https://branchlab.github.io/metasnf/articles/clustering_algorithms.html.
- A column named "cont_dist": Specification of which distance metric will be used for dataframes of purely continuous data. You can learn about this metric and its defaults here: https://branchlab.github.io/metasnf/artic
- A column named "disc_dist": Like above, but for discrete dataframes.
- A column named "ord_dist": Like above, but for ordinal dataframes.
- A column named "cat_dist": Like above, but for categorical dataframes.
- A column named "mixed_dist": Like above, but for mixed-type (e.g., both categorical and discrete) dataframes.

- One column for every input dataframe in the corresponding data_list which can either have the value of 0 or 1. The name of the column should be formatted as "inc_[]" where the square brackets are replaced with the name (as found in dl_summary(data_list)$"name") of each dataframe. When 0, that dataframe will be excluded from that run of the SNF pipeline. When 1, that dataframe will be included.

Usage

```
generate_settings_matrix(
  data_list,
  seed = NULL,
  nrows = 0,
  min_removed_inputs = 0,
  max_removed_inputs = length(data_list) - 1,
  dropout_dist = "exponential",
  min_alpha = NULL,
  max_alpha = NULL,
  min_k = NULL,
  max_k = NULL,
  min_t = NULL,
  max_t = NULL,
  alpha_values = NULL,
  k_values = NULL,
  t_values = NULL,
  possible_snf_schemes = c(1, 2, 3),
  clustering_algorithms = NULL,
  continuous_distances = NULL,
  discrete_distances = NULL,
  ordinal_distances = NULL,
  categorical_distances = NULL,
  mixed_distances = NULL,
  distance_metrics_list = NULL,
  snf_input_weights = NULL,
  snf_domain_weights = NULL,
  retry_limit = 10
)
```

Arguments

data_list	A nested list of input data from `generate_data_list()`.
seed	(DEPRECATED) set the global seed. To ensure reproducible settings matrices are generated, manually call `set.seed()` prior to settings matrix generation instead of using this parameter.
nrows	Number of rows to generate for the settings matrix.
min_removed_inputs	The smallest number of input dataframes that may be randomly removed. By default, 0.

`max_removed_inputs`
: The largest number of input dataframes that may be randomly removed. By default, this is 1 less than all the provided input dataframes in the data_list.

`dropout_dist`
: Parameter controlling how the random removal of input dataframes should occur. Can be "none" (no input dataframes are randomly removed), "uniform" (uniformly sample between min_removed_inputs and max_removed_inputs to determine number of input dataframes to remove), or "exponential" (pick number of input dataframes to remove by sampling from min_removed_inputs to max_removed_inputs with an exponential distribution; the default).

`min_alpha`
: The minimum value that the alpha hyperparameter can have. Random assigned value of alpha for each row will be obtained by uniformly sampling numbers between `min_alpha` and `max_alpha` at intervals of 0.1. Cannot be used in conjunction with the `alpha_values` parameter.

`max_alpha`
: The maximum value that the alpha hyperparameter can have. See `min_alpha` parameter. Cannot be used in conjunction with the `alpha_values` parameter.

`min_k`
: The minimum value that the k hyperparameter can have. Random assigned value of k for each row will be obtained by uniformly sampling numbers between `min_k` and `max_k` at intervals of 1. Cannot be used in conjunction with the `k_values` parameter.

`max_k`
: The maximum value that the k hyperparameter can have. See `min_k` parameter. Cannot be used in conjunction with the `k_values` parameter.

`min_t`
: The minimum value that the t hyperparameter can have. Random assigned value of t for each row will be obtained by uniformly sampling numbers between `min_t` and `max_t` at intervals of 1. Cannot be used in conjunction with the `t_values` parameter.

`max_t`
: The maximum value that the t hyperparameter can have. See `min_t` parameter. Cannot be used in conjunction with the `t_values` parameter.

`alpha_values`
: A number or numeric vector of a set of possible values that alpha can take on. Value will be obtained by uniformly sampling the vector. Cannot be used in conjunction with the `min_alpha` or `max_alpha` parameters.

`k_values`
: A number or numeric vector of a set of possible values that k can take on. Value will be obtained by uniformly sampling the vector. Cannot be used in conjunction with the `min_k` or `max_k` parameters.

`t_values`
: A number or numeric vector of a set of possible values that t can take on. Value will be obtained by uniformly sampling the vector. Cannot be used in conjunction with the `min_t` or `max_t` parameters.

`possible_snf_schemes`
: A vector containing the possible snf_schemes to uniformly randomly select from. By default, the vector contains all 3 possible schemes: c(1, 2, 3). 1 corresponds to the "individual" scheme, 2 corresponds to the "domain" scheme, and 3 corresponds to the "twostep" scheme.

`clustering_algorithms`
: A list of clustering algorithms to uniformly randomly pick from when clustering. When not specified, randomly select between spectral clustering using the eigen-gap heuristic and spectral clustering using the rotation cost heuristic. See

?generate_clust_algs_list for more details on running custom clustering algorithms.

`continuous_distances`
: A vector of continuous distance metrics to use when a custom distance_metrics_list is provided.

`discrete_distances`
: A vector of categorical distance metrics to use when a custom distance_metrics_list is provided.

`ordinal_distances`
: A vector of categorical distance metrics to use when a custom distance_metrics_list is provided.

`categorical_distances`
: A vector of categorical distance metrics to use when a custom distance_metrics_list is provided.

`mixed_distances`
: A vector of mixed distance metrics to use when a custom distance_metrics_list is provided.

`distance_metrics_list`
: List containing distance metrics to vary over. See ?generate_distance_metrics_list.

`snf_input_weights`
: Nested list containing weights for when SNF is used to merge individual input measures (see ?generate_snf_weights)

`snf_domain_weights`
: Nested list containing weights for when SNF is used to merge domains (see ?generate_snf_weights)

`retry_limit`
: The maximum number of attempts to generate a novel row. This function does not return matrices with identical rows. As the range of requested possible settings tightens and the number of requested rows increases, the risk of randomly generating a row that already exists increases. If a new random row has matched an existing row `retry_limit` number of times, the function will terminate.

Value

settings_matrix A settings matrix

`generate_weights_matrix`
Generate a matrix to store feature weights

Description

Generate a matrix to store feature weights

Usage

```
generate_weights_matrix(data_list = NULL, nrow = 1, fill = "ones")
```

Arguments

data_list	A nested list of input data from generate_data_list().
nrow	Number of rows to generate the template weights matrix for.
fill	String indicating what to populate generate rows with. Can be "ones" (default; fill matrix with 1), "uniform" (fill matrix with uniformly distributed random values), or "exponential" (fill matrix with exponentially distributed random values).

Value

weights_matrix A properly formatted matrix containing columns for all the features that require weights and rows.

---

get_clusters *Extract cluster membership vector from one solutions matrix row*

Description

This function takes in a single row of a solutions matrix and returns a vector containing the cluster assignments for each observation. It is similar to get_cluster_df(), which takes a solutions matrix with only one row and returns a dataframe with two columns: "cluster" and "subjectkey" '(the UID of the observation) and get_cluster_solutions(), which takes a solutions matrix with any number of rows and returns a dataframe indicating the cluster assignments for each of those rows.

Usage

```
get_clusters(solutions_matrix_row)
```

Arguments

solutions_matrix_row	Output matrix row.

Value

clusters Vector of assigned clusters.

get_cluster_df *Extract cluster membership information from one solutions matrix row*

Description

This function takes in a single row of a solutions matrix and returns a dataframe containing the cluster assignments for each subjectkey. It is similar to `get_clusters()`, which takes one solutions matrix row and returns a vector of cluster assignments' and `get_cluster_solutions()`, which takes a solutions matrix with any number of rows and returns a dataframe indicating the cluster assignments for each of those rows.

Usage

```
get_cluster_df(solutions_matrix_row)
```

Arguments

`solutions_matrix_row`
: One row from a solutions matrix.

Value

cluster_df dataframe of cluster and subjectkey.

get_cluster_solutions *Extract cluster membership information from a solutions_matrix*

Description

This function takes in a solutions matrix and returns a dataframe containing the cluster assignments for each subjectkey. It is similar to 'get_clusters(), which takes one solutions matrix row and returns a vector of cluster assignments' and `get_cluster_df()`, which takes a solutions matrix with only one row and returns a dataframe with two columns: "cluster" and "subjectkey" (the UID of the observation).

Usage

```
get_cluster_solutions(solutions_matrix)
```

Arguments

`solutions_matrix`
: A solutions_matrix.

Value

cluster_solutions A "data.frame" object where each row is an observation and each column (apart from the subjectkey column) indicates the cluster that observation was assigned to for the corresponding solutions matrix row.

`get_complete_uids` *Pull complete-data UIDs from a list of dataframes*

Description

Pull complete-data UIDs from a list of dataframes

Usage

```
get_complete_uids(list_of_dfs, uid)
```

Arguments

`list_of_dfs`	List of dataframes.
`uid`	Name of column across dataframes containing UIDs

Value

A character vector of the UIDs of observations that have complete data across the provided list of dataframes.

`get_dist_matrix` *Calculate distance matrices*

Description

Given a dataframe of numerical features, return a euclidean distance matrix.

Usage

```
get_dist_matrix(
  df,
  input_type,
  cont_dist_fn,
  disc_dist_fn,
  ord_dist_fn,
  cat_dist_fn,
  mix_dist_fn,
  weights_row
)
```

Arguments

df	Raw dataframe with subject IDs in column "subjectkey"
input_type	Either "numeric" (resulting in euclidean distances), "categorical" (resulting in binary distances), or "mixed" (resulting in gower distances)
cont_dist_fn	distance metric function for continuous data
disc_dist_fn	distance metric function for discrete data
ord_dist_fn	distance metric function for ordinal data
cat_dist_fn	distance metric function for categorical data
mix_dist_fn	distance metric function for mixed data
weights_row	Single-row dataframe where the column names contain the column names in df and the row contains the corresponding weights_row.

Value

dist_matrix Matrix of inter-observation distances.

---

get_dl_subjects *Extract subjects from a data_list*

Description

Extract subjects from a data_list

Usage

```
get_dl_subjects(data_list, prefix = FALSE)
```

Arguments

data_list	A nested list of input data from `generate_data_list()`.
prefix	If TRUE, preserves the "subject_" prefix added to UIDs when creating a data_list.

Value

A character vector of the UID labels contained in a data list.

get_heatmap_order *Return the row or column ordering present in a heatmap*

Description

Return the row or column ordering present in a heatmap

Usage

```
get_heatmap_order(heatmap, type = "rows")
```

Arguments

heatmap	A heatmap object to collect ordering from.
type	The type of ordering to return. Either "rows" or "columns".

Value

A numeric vector of the ordering used within the provided ComplexHeatmap "Heatmap" object.

get_matrix_order *Return the hierarchical clustering order of a matrix*

Description

Return the hierarchical clustering order of a matrix

Usage

```
get_matrix_order(matrix, dist_method = "euclidean", hclust_method = "complete")
```

Arguments

matrix	Matrix to cluster.
dist_method	Distance method to apply to the matrix. Argument is directly passed into stats::dist. Options include "euclidean", "maximum", "manhattan", "canberra", "binary", or "minkowski".
hclust_method	Which agglomerative method to be passed into stats::hclust. Options include "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median", or "centroid".

Value

A numeric vector of the ordering derivied by the specified hierarchical clustering method applied to the provided matrix.

get_mean_pval *Get mean p-value*

Description

Given an solutions matrix row containing evaluated p-values, returns mean.

Usage

```
get_mean_pval(solutions_matrix_row)
```

Arguments

solutions_matrix_row	row of solutions_matrix object

Value

mean_pval mean p-value

get_min_pval *Get minimum p-value*

Description

Given an solutions matrix row containing evaluated p-values, returns min.

Usage

```
get_min_pval(solutions_matrix_row)
```

Arguments

solutions_matrix_row	row of solutions_matrix object

Value

min_pval minimum p-value

get_pvals *Get p-values from an extended solutions matrix*

Description

This function can be used to neatly format the p-values associated with an extended solutions matrix. It can also calculate the negative logs of those p-values to make it easier to interpret large-scale differences.

Usage

```
get_pvals(
  extended_solutions_matrix,
  negative_log = FALSE,
  keep_summaries = TRUE
)
```

Arguments

extended_solutions_matrix	The output of extend_solutions. A dataframe that contains at least one p-value column ending in "_pval".
negative_log	If TRUE, will replace p-values with negative log p-values.
keep_summaries	If FALSE, will remove the mean, min, and max p-value.

Value

A "data.frame" class object Of only the p-value related columns of the provided extended_solutions_matrix.

get_representative_solutions *Extract representative solutions from a matrix of ARIs*

Description

Following clustering with batch_snf, a matrix of pairwise ARIs that show how related each cluster solution is to each other can be generated by the calc_aris function. Partitioning of the ARI matrix can be done by visual inspection of adjusted_rand_index_heatmap() or by shiny_annotator. Given the indices of meta cluster boundaries, this function will return a single representative solution from each meta cluster based on maximum average ARI to all other solutions within that meta cluster.

Usage

```
get_representative_solutions(
  aris,
  split_vector,
  order,
  solutions_matrix,
  filter_fn = NULL
)
```

Arguments

aris	Matrix of adjusted rand indices from calc_aris()
split_vector	A vector of partition indices.
order	Numeric vector indicating row ordering of settings matrix.
solutions_matrix	Output of batch_snf containing cluster solutions.
filter_fn	Optional function to filter the meta-cluster by prior to maximum average ARI determination. This can be useful if you are explicitly trying to select a solution that meets a certain condition, such as only picking from the 4 cluster solutions within a meta cluster. An example valid function could be fn <- function(x) x[x$"nclust" == 4,].

Value

A "data.frame" class object corresponding to a subset of the provided solutions matrix in which only one row is present per meta cluster.

---

gower_distance *Distance metric: Gower distance*

Description

Distance metric: Gower distance

Usage

```
gower_distance(df, weights_row)
```

Arguments

df	Dataframe containing at least 1 data column.
weights_row	For compatibility - function does not accept weights.

Value

distance_matrix A distance matrix.

hamming_distance *Distance metric: Hamming distance*

Description

Distance metric: Hamming distance

Usage

```
hamming_distance(df, weights_row)
```

Arguments

df	Dataframe containing one subjectkey column in the first column and at least 1 categorical data column. All feature data should be categorical.
weights_row	Single-row dataframe where the column names contain the column names in df and the row contains the corresponding weights.

Value

distance_matrix A distance matrix.

income *Mock ABCD income data*

Description

Like the mock dataframe "abcd_h_income", but with "unique_id" as the "uid".

Like the mock dataframe "abcd_cort_sa", but with "unique_id" as the "uid".

Usage

```
income

income
```

Format

`income`**:**
A data frame with 300 rows and 2 columns:
unique_id The unique identifier of the ABCD dataset
household_income Household income in 3 category levels (low = 1, medium = 2, high = 3)

`income`**:**
A data frame with 300 rows and 2 columns:
unique_id The unique identifier of the ABCD dataset
household_income Household income in 3 category levels (low = 1, medium = 2, high = 3)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

`individual` *SNF Scheme: Individual*

Description

The "vanilla" scheme - does distance matrix conversions of each input dataframe in a list and

Usage

```
individual(
  data_list,
  cont_dist_fn,
```

```
  disc_dist_fn,
  ord_dist_fn,
  cat_dist_fn,
  mix_dist_fn,
  weights_row,
  k,
  alpha,
  t
)
```

Arguments

data_list	A nested list of input data from generate_data_list().
cont_dist_fn	distance metric function for continuous data.
disc_dist_fn	distance metric function for discrete data.
ord_dist_fn	distance metric function for ordinal data.
cat_dist_fn	distance metric function for categorical data.
mix_dist_fn	distance metric function for mixed data.
weights_row	dataframe row containing feature weights.
k	k hyperparameter.
alpha	alpha/eta/sigma hyperparameter.
t	SNF number of iterations hyperparameter.

Value

fused_network The final fused network (class "matrix", "array") generated by SNF.

---

jitter_plot *Jitter plot separating a feature by cluster*

Description

Jitter plot separating a feature by cluster

Usage

```
jitter_plot(df, feature)
```

Arguments

df	A data.frame containing cluster column and the feature to plot.
feature	The feature to plot.

Value

A jitter+violin plot (class "gg", "ggplot") showing the distribution of a feature across clusters.

label_prop *Label propagation*

Description

Given a full fused network (one containing both pre-labeled training subjects and unlabeled test-subjects) and the clusters of the pre-labeled subjects, return a label propagated list of clusters for all subjects. This function is derived from SNFtool::groupPredict. Modifications are made to take a full fused network as input, rather than taking input dataframes and running SNF internally. This ensures that alternative approaches to data normalization and distance matrix calculations can be chosen by the user.

Usage

```
label_prop(full_fused_network, clusters)
```

Arguments

full_fused_network	Network made by running SNF on training and test subjects together
clusters	a vector of training subject assigned clusters in matching order as they appear in full_fused_network

Value

new_clusters list of cluster labels for all subjects

label_splits *Convert a vector of partition indices into meta cluster labels*

Description

Convert a vector of partition indices into meta cluster labels

Usage

```
label_splits(split_vector, nrow)
```

Arguments

split_vector	A vector of partition indices.
nrow	The number of rows in the data being partitioned.

Value

A character vector that expands the split_vector into an nrow-length sequence of ascending letters of the alphabet. If the split vector is c(3, 6) and the number of rows is 8, the result will be a vector of two "A"s (up to the first index, 3), three "B"s (up to the second index, 6), and three "C"s (up to and including the last index, 8).

---

linear_adjust *Linearly correct data_list by features with unwanted signal*

Description

Given a data_list to correct and another data_list of categorical features to linearly adjust for, corrects the first data_list based on the residuals of the linear model relating the numeric features in the first data_list to the unwanted signal features in the second data list.

Usage

```
linear_adjust(data_list, unwanted_signal_list, sig_digs = NULL)
```

Arguments

data_list	A nested list of input data from generate_data_list().
unwanted_signal_list	A data_list of categorical features that should have their mean differences removed in the first data_list.
sig_digs	Number of significant digits to round the residuals to.

Value

A data list ("list") in which each data component has been converted to contain residuals off of the linear model built against the features in the unwanted_signal_list.

---

linear_model_pval *Linear model p-value (generic)*

Description

Return p-value of F-test for a linear model of any two features

Usage

```
linear_model_pval(predictor, response)
```

Arguments

predictor	A categorical or numeric feature.
response	A numeric feature.

Value

pval A p-value (class "numeric").

---

list_remove *Remove items from a data_list*

Description

Removes specified elements from a provided data_list

Usage

```
list_remove(list_object, ...)
```

Arguments

list_object	The data_list containing components to be removed
...	Any number of components to remove from the list object, passed as strings

Value

A "list"-class object in which any specified elements have been removed.

---

lp_solutions_matrix *Label propagate cluster solutions to unclustered subjects*

Description

Given a solutions_matrix derived from training subjects and a full_data_list containing both training and test subjects, re-run SNF to generate a total affinity matrix of both train and subjects and use the label propagation algorithm to assigned predicted clusters to test subjects.

Usage

```
lp_solutions_matrix(
  train_solutions_matrix,
  full_data_list,
  distance_metrics_list = NULL,
  weights_matrix = NULL,
  verbose = FALSE
)
```

Arguments

`train_solutions_matrix`
: A solutions_matrix derived from the training set. The propagation algorithm is slow and should be used for validating a top or top few meaningful chosen clustering solutions. It is advisable to use only a small subset of rows from the original training solutions_matrix for label propagation.

`full_data_list`
: A data_list containing subjects from both the training and testing sets.

`distance_metrics_list`
: Like above - the distance_metrics_list (if any) that was used for the original batch_snf call.

`weights_matrix`
: Like above.

`verbose`
: If TRUE, print progress to console.

Value

labeled_df a dataframe containing a column for subjectkeys, a column for whether the subject was in the train (original) or test (held out) set, and one column per row of the solutions matrix indicating the original and propagated clusters.

`mc_manhattan_plot` *Manhattan plot of feature-meta cluster associaiton p-values*

Description

Given a dataframe of representative meta cluster solutions (see `get_representative_solutions()`, returns a Manhattan plot for showing feature separation across all features in provided data/target_lists.

Usage

```
mc_manhattan_plot(
  extended_solutions_matrix,
  data_list = NULL,
  target_list = NULL,
  variable_order = NULL,
  neg_log_pval_thresh = 5,
  threshold = NULL,
  point_size = 5,
  text_size = 20,
  plot_title = NULL,
  xints = NULL,
  hide_x_labels = FALSE,
  domain_colours = NULL
)
```

Arguments

extended_solutions_matrix	A solutions_matrix that contains "_pval" columns containing the values to be plotted. This object is the output of extend_solutions().
data_list	List of dataframes containing data information.
target_list	List of dataframes containing target information.
variable_order	Order of features to be displayed in the plot.
neg_log_pval_thresh	Threshold for negative log p-values.
threshold	p-value threshold to plot horizontal dashed line at.
point_size	Size of points in the plot.
text_size	Size of text in the plot.
plot_title	Title of the plot.
xints	Either "outcomes" or a vector of numeric values to plot vertical lines at.
hide_x_labels	If TRUE, hides x-axis labels.
domain_colours	Named vector of colours for domains.

Value

A Manhattan plot (class "gg", "ggplot") showing the association p-values of features against each solution in the provided solutions matrix, stratified by meta cluster label.

---

merge_data_lists *Horizontally merge compatible data lists*

Description

Join two data_lists with the same components (dataframes) but separate observations. To instead merge two data_lists that have the same observations but different components, simply use c().

Usage

```
merge_data_lists(data_list1, data_list2)
```

Arguments

data_list1	The first data_list to merge.
data_list2	The second data_list to merge.

Value

A data list ("list"-class object) containing the observations of both provided data lists.

merge_df_list *Merge list of dataframes*

Description

Merge list of dataframes

Usage

```
merge_df_list(df_list, join = "inner", uid = "subjectkey", no_na = FALSE)
```

Arguments

df_list	list of dataframes
join	String indicating if join should be "inner" or "full"
uid	Column name to join on. Default is "subjectkey"
no_na	Whether to remove NA values from the merged dataframe

Value

merged_df inner join of all dataframes in list

methylation_df *Modification of SNFtool mock dataframe "Data2"*

Description

Modification of SNFtool mock dataframe "Data2"

Usage

```
methylation_df
```

Format

`methylation_df`**:**
A data frame with 200 rows and 3 columns:

gene_1_expression Mock gene methylation feature

gene_2_expression Mock gene methylation feature

patient_id Random three-digit number uniquely identifying the patient

Source

This data came from the SNFtool package, with slight modifications.

no_subs *Select all columns of a dataframe not starting with the 'subject_' prefix.*

Description

Removes the 'subject_' prefixed columns from a dataframe. Useful for printing solutions_matrix structures to the console.

Usage

```
no_subs(df)
```

Arguments

df	A dataframe

Value

df_no_subs Dataframe without subjects

numcol_to_numeric *Convert dataframe columns to numeric type*

Description

Converts all columns in a dataframe that can be converted to numeric type to numeric type.

Usage

```
numcol_to_numeric(df)
```

Arguments

df	A dataframe

Value

df The dataframe with all possible columns converted to type numeric

ord_reg_pval *Ordinal regression p-value*

Description

Returns the overall p-value of an ordinal regression on a categorical predictor and response vetors. If the ordinal response

Usage

```
ord_reg_pval(predictor, response)
```

Arguments

predictor	A categorical or numeric feature.
response	A numeric feature.

Value

pval A p-value (class "numeric").

parallel_batch_snf *Parallel processing form of batch_snf*

Description

Parallel processing form of batch_snf

Usage

```
parallel_batch_snf(
  data_list,
  distance_metrics_list,
  clust_algs_list,
  settings_matrix,
  weights_matrix,
  similarity_matrix_dir,
  return_similarity_matrices,
  processes
)
```

Arguments

data_list	A nested list of input data from generate_data_list().
distance_metrics_list	An optional nested list containing which distance metric function should be used for the various feature types (continuous, discrete, ordinal, categorical, and mixed). See ?generate_distance_metrics_list for details on how to build this.
clust_algs_list	List of custom clustering algorithms to apply to the final fused network. See ?generate_clust_algs_list.
settings_matrix	matrix indicating parameters to iterate SNF through.
weights_matrix	A matrix containing feature weights to use during distance matrix calculation. See ?generate_weights_matrix for details on how to build this.
similarity_matrix_dir	If specified, this directory will be used to save all generated similarity matrices.
return_similarity_matrices	If TRUE, function will return a list where the first element is the solutions matrix and the second element is a list of similarity matrices for each row in the solutions_matrix. Default FALSE.
processes	Number of parallel processes used when executing SNF.

Value

The same values as ?batch_snf().

---

prefix_dl_sk *Add "subject_" prefix to all UID values in subjectkey column*

Description

Add "subject_" prefix to all UID values in subjectkey column

Usage

```
prefix_dl_sk(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

data_list A data_list without NAs

pubertal *Mock ABCD pubertal status data*

Description

Like the mock dataframe "abcd_pubertal", but with "unique_id" as the "uid".

Usage

```
pubertal
```

Format

`pubertal`**:**

A data frame with 275 rows and 2 columns:

unique_id The unique identifier of the ABCD dataset

pubertal_status Average reported pubertal status between child and parent (1-5 categorical scale)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

pval_heatmap *Heatmap of p-values*

Description

Heatmap of p-values

Usage

```
pval_heatmap(
  pvals,
  order = NULL,
  cluster_columns = TRUE,
  cluster_rows = FALSE,
  show_row_names = FALSE,
  show_column_names = TRUE,
  min_colour = "red2",
  mid_colour = "lightyellow",
  max_colour = "slateblue4",
  legend_breaks = c(0, 0.5, 1),
 col = circlize::colorRamp2(legend_breaks, c(min_colour, mid_colour, max_colour)),
 heatmap_legend_param = list(color_bar = "continuous", title = "p-value", at = c(0, 0.5,
    1)),
  rect_gp = grid::gpar(col = "black"),
  column_split_vector = NULL,
  row_split_vector = NULL,
  column_split = NULL,
  row_split = NULL,
  ...
)
```

Arguments

pvals	A matrix of p-values.
order	Numeric vector containing row order of the heatmap.
cluster_columns	Whether columns should be sorted by hierarchical clustering.
cluster_rows	Whether rows should be sorted by hierarchical clustering.
show_row_names	Whether row names should be shown.
show_column_names	Whether column names should be shown.
min_colour	Colour used for the lowest value in the heatmap.
mid_colour	Colour used for the middle value in the heatmap.
max_colour	Colour used for the highest value in the heatmap.
legend_breaks	Numeric vector of breaks for the legend.
col	Colour function for `ComplexHeatmap::Heatmap()`
heatmap_legend_param	Legend function for `ComplexHeatmap::Heatmap()`
rect_gp	Cell border function for `ComplexHeatmap::Heatmap()`
column_split_vector	Vector of indices to split columns by.
row_split_vector	Vector of indices to split rows by.

column_split	Standard parameter of ComplexHeatmap::Heatmap.
row_split	Standard parameter of ComplexHeatmap::Heatmap.
...	Additional parameters passed to ComplexHeatmap::Heatmap.

Value

Returns a heatmap (class "Heatmap" from package ComplexHeatmap) that displays the provided p-values.

---

random_removal *Generate random removal sequence*

Description

Helper function to contribute to rows within the settings matrix. Number of columns removed follows a uniform or exponential probability distribution.

Usage

```
random_removal(
  columns,
  min_removed_inputs,
  max_removed_inputs,
  dropout_dist = "exponential"
)
```

Arguments

columns	Columns of the settings_matrix that are passed in
min_removed_inputs	The smallest number of input dataframes that may be randomly removed.
max_removed_inputs	The largest number of input dataframes that may be randomly removed.
dropout_dist	Indication of how input dataframes should be dropped. can be "none" (no dropout), "uniform" (uniformly draw number between min and max removed inputs), or "exponential" (like uniform, but using an exponential distribution; default).

Value

inclusions_df Dataframe that can be rbind'ed to the settings_matrix

reduce_dl_to_common *Reduce data_list to common subjects*

Description

Given a data_list object, reduce each nested dataframe to contain only the set of subjects that are shared by all nested dataframes

Usage

```
reduce_dl_to_common(data_list)
```

Arguments

data_list	A nested list of input data from generate_data_list().

Value

reduced_data_list The data_list object subsetted only to subjectssnf shared across all nested dataframes

remove_dl_na *Remove NAs from a data_list object*

Description

Remove NAs from a data_list object

Usage

```
remove_dl_na(data_list, return_missing = FALSE)
```

Arguments

data_list	A nested list of input data from generate_data_list().
return_missing	If TRUE, function returns a list where the first element is the data_list and the second element is a vector of unique IDs of patients who were removed during the complete data filtration step.

Value

data_list A data_list without NAs

rename_dl *Rename features in a data_list*

Description

Rename features in a data_list

Usage

```
rename_dl(data_list, name_mapping)
```

Arguments

data_list	A nested list of input data from generate_data_list().
name_mapping	A named vector where the values are the features to be renamed and the names are the new names for those features.

Value

A data list ("list"-class object) with adjusted feature names.

Examples

```
library(metasnf)

data_list <- generate_data_list(
    list(pubertal, "pubertal_status", "demographics", "continuous"),
    list(anxiety, "anxiety", "behaviour", "ordinal"),
    list(depress, "depressed", "behaviour", "ordinal"),
    uid = "unique_id"
)

summarize_dl(data_list, "feature")

name_changes <- c(
    "anxiety_score" = "cbcl_anxiety_r",
    "depression_score" = "cbcl_depress_r"
)

data_list <- rename_dl(data_list, name_changes)

summarize_dl(data_list, "feature")
```

reorder_dl_subs *Reorder the subjects in a data_list*

Description

Reorder the subjects in a data_list

Usage

```
reorder_dl_subs(data_list, ordered_subjects)
```

Arguments

data_list	A nested list of input data from generate_data_list().
ordered_subjects	A vector of the subjectkey values in the data_list in the desired order of the sorted data_list.

Value

A data list ("list"-class object) with reordered observations.

resample *Helper resample function found in ?sample*

Description

Like sample, but when given a single value x, returns back that single value instead of a random value from 1 to x.

Usage

```
resample(x, ...)
```

Arguments

x	Vector or single value to sample from
...	Remaining arguments for base::sample function

Value

Numeric vector result of running base::sample.

save_heatmap *Save a heatmap object to a file*

Description

Save a heatmap object to a file

Usage

```
save_heatmap(heatmap, path, width = 480, height = 480, res = 100)
```

Arguments

heatmap	The heatmap object to save.
path	The path to save the heatmap to.
width	The width of the heatmap.
height	The height of the heatmap.
res	The resolution of the heatmap.

Value

Does not return any value. Saves heatmap to file.

scale_diagonals *Adjust the diagonals of a matrix*

Description

Adjust the diagonals of a matrix to reduce contrast with off-diagonals during plotting.

Usage

```
scale_diagonals(matrix, method = "mean")
```

Arguments

matrix	Matrix to rescale.
method	Method of rescaling. Can be: • "mean" (replace diagonals with average value of off-diagonals) • "zero" (replace diagonals with 0) • "min" (replace diagonals with min value of off-diagonals) • "max" (replace diagonals with max value of off-diagonals)

Value

A "matrix" class object with rescaled diagonals.

`settings_matrix_heatmap` *Heatmap for visualizing a settings matrix*

Description

Scales settings matrix values between 0 and 1 and plots as a heatmap. Rows can be reordered to match prior meta clustering results.

Usage

```
settings_matrix_heatmap(
  settings_matrix,
  order = NULL,
  remove_fixed_columns = TRUE,
  show_column_names = TRUE,
  show_row_names = TRUE,
  rect_gp = grid::gpar(col = "black"),
  colour_breaks = c(0, 1),
  colours = c("black", "darkseagreen"),
  column_split_vector = NULL,
  row_split_vector = NULL,
  column_split = NULL,
  row_split = NULL,
  column_title = NULL,
  ...
)
```

Arguments

`settings_matrix`	Matrix indicating parameters to iterate SNF through.
`order`	Numeric vector indicating row ordering of settings matrix.
`remove_fixed_columns`	Whether columns that have no variation should be removed.
`show_column_names`	Whether column names should be shown.
`show_row_names`	Whether row names should be shown.
`rect_gp`	Cell border function for `ComplexHeatmap::Heatmap`.
`colour_breaks`	Numeric vector of breaks for the legend.
`colours`	Vector of colours to use for the heatmap. Should match the length of colour_breaks.
`column_split_vector`	Vector of indices to split columns by.
`row_split_vector`	Vector of indices to split rows by.

column_split	Standard parameter of ComplexHeatmap::Heatmap.
row_split	Standard parameter of ComplexHeatmap::Heatmap.
column_title	Standard parameter of ComplexHeatmap::Heatmap.
...	Additional parameters passed to ComplexHeatmap::Heatmap.

Value

Returns a heatmap (class "Heatmap" from package ComplexHeatmap) that displays the scaled values of the provided settings matrix.

---

sew_euclidean_distance *Squared (excluding weights) Euclidean distance*

Description

Squared (excluding weights) Euclidean distance

Usage

```
sew_euclidean_distance(df, weights_row)
```

Arguments

df	Dataframe containing at least 1 data column.
weights_row	Single-row dataframe where the column names contain the column names in df and the row contains the corresponding weights.

Value

distance_matrix A distance matrix.

---

shiny_annotator *Launch shiny app to identify meta cluster boundaries*

Description

Launch shiny app to identify meta cluster boundaries

Usage

```
shiny_annotator(ari_heatmap)
```

Arguments

ari_heatmap	Heatmap of ARIs to divide into meta clusters.

Value

Does not return any value. Launches interactive shiny applet.

`similarity_matrix_heatmap` *Plot heatmap of similarity matrix*

Description

Plot heatmap of similarity matrix

Usage

```
similarity_matrix_heatmap(
  similarity_matrix,
  order = NULL,
  cluster_solution = NULL,
  scale_diag = "mean",
  log_graph = TRUE,
  cluster_rows = FALSE,
  cluster_columns = FALSE,
  show_row_names = FALSE,
  show_column_names = FALSE,
  data_list = NULL,
  data = NULL,
  left_bar = NULL,
  right_bar = NULL,
  top_bar = NULL,
  bottom_bar = NULL,
  left_hm = NULL,
  right_hm = NULL,
  top_hm = NULL,
  bottom_hm = NULL,
  annotation_colours = NULL,
  min_colour = NULL,
  max_colour = NULL,
  split_vector = NULL,
  row_split = NULL,
  column_split = NULL,
  ...
)
```

Arguments

`similarity_matrix` A similarity matrix

order	Vector of numbers to reorder the similarity matrix (and data if provided). Overwrites ordering specified by cluster_solution param.
cluster_solution	Vector containing cluster assignments.
scale_diag	Method of rescaling matrix diagonals. Can be "none" (don't change diagonals), "mean" (replace diagonals with average value of off-diagonals), or "zero" (replace diagonals with 0).
log_graph	If TRUE, log transforms the graph.
cluster_rows	Parameter for ComplexHeatmap::Heatmap.
cluster_columns	Parameter for ComplexHeatmap::Heatmap.
show_row_names	Parameter for ComplexHeatmap::Heatmap.
show_column_names	Parameter for ComplexHeatmap::Heatmap.
data_list	A nested list of input data from `generate_data_list()`.
data	A dataframe containing elements requested for annotation.
left_bar	Named list of strings, where the strings are features in df that should be used for a barplot annotation on the left of the plot and the names are the names that will be used to caption the plots and their legends.
right_bar	See left_bar.
top_bar	See left_bar.
bottom_bar	See left_bar.
left_hm	Like left_bar, but with a heatmap annotation instead of a barplot annotation.
right_hm	See left_hm.
top_hm	See left_hm.
bottom_hm	See left_hm.
annotation_colours	Named list of heatmap annotations and their colours.
min_colour	Colour used for the lowest value in the heatmap.
max_colour	Colour used for the highest value in the heatmap.
split_vector	A vector of partition indices.
row_split	Standard parameter of `ComplexHeatmap::Heatmap`.
column_split	Standard parameter of `ComplexHeatmap::Heatmap`.
...	Additional parameters passed into ComplexHeatmap::Heatmap.

Value

Returns a heatmap (class "Heatmap" from package ComplexHeatmap) that displays the similarities between observations in the provided matrix.

`similarity_matrix_path` *Generate a complete path and filename to store an similarity matrix*

Description

Generate a complete path and filename to store an similarity matrix

Usage

```
similarity_matrix_path(similarity_matrix_dir, i)
```

Arguments

`similarity_matrix_dir`	Directory to store similarity matrices
`i`	Corresponding settings matrix row

Value

path Complete path and filename to store an similarity matrix

---

`siw_euclidean_distance` *Squared (including weights) Euclidean distance*

Description

Squared (including weights) Euclidean distance

Usage

```
siw_euclidean_distance(df, weights_row)
```

Arguments

`df`	Dataframe containing at least 1 data column.
`weights_row`	Single-row dataframe where the column names contain the column names in df and the row contains the corresponding weights.

Value

distance_matrix A distance matrix.

`snf_step` *Convert a data list to a similarity matrix through a variety of SNF schemes*

Description

Convert a data list to a similarity matrix through a variety of SNF schemes

Usage

```
snf_step(
  data_list,
  scheme,
  k = 20,
  alpha = 0.5,
  t = 20,
  cont_dist_fn,
  disc_dist_fn,
  ord_dist_fn,
  cat_dist_fn,
  mix_dist_fn,
  weights_row
)
```

Arguments

`data_list`	A nested list of input data from `generate_data_list()`.
`scheme`	Which SNF system to use to achieve the final fused network.
`k`	k hyperparameter.
`alpha`	alpha/eta/sigma hyperparameter.
`t`	SNF number of iterations hyperparameter.
`cont_dist_fn`	distance metric function for continuous data.
`disc_dist_fn`	distance metric function for discrete data.
`ord_dist_fn`	distance metric function for ordinal data.
`cat_dist_fn`	distance metric function for categorical data.
`mix_dist_fn`	distance metric function for mixed data.
`weights_row`	dataframe row containing feature weights.

Value

fused_network The final fused network (class "matrix", "array") generated by SNF.

sn_euclidean_distance *Distance metric: Standard normalization then Euclidean*

Description

Distance metric: Standard normalization then Euclidean

Usage

```
sn_euclidean_distance(df, weights_row)
```

Arguments

df	Dataframe containing at least 1 data column.
weights_row	Single-row dataframe where the column names contain the column names in df and the row contains the corresponding weights.

Value

distance_matrix A distance matrix.

spectral_eigen *Clustering algorithm: Spectral clustering with eigen-gap heuristic*

Description

Applies spectral clustering to similarity matrix. Number of clusters is based on the eigen-gap heuristic.

Usage

```
spectral_eigen(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_eigen_classic *Clustering algorithm: Spectral clustering with eigen-gap heuristic*

Description

Applies spectral clustering to similarity matrix. Number of clusters is based on the eigen-gap heuristic. Range of possible cluster solutions is fixed between 2 and 5 inclusive.

Usage

```
spectral_eigen_classic(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_eight *Clustering algorithm: Spectral clustering for a eight cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks eight clusters.

Usage

```
spectral_eight(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_five *Clustering algorithm: Spectral clustering for a five cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks five clusters.

Usage

```
spectral_five(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_four *Clustering algorithm: Spectral clustering for a four cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks four clusters.

Usage

```
spectral_four(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_nine *Clustering algorithm: Spectral clustering for a nine cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks nine clusters.

Usage

```
spectral_nine(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_rot *Clustering algorithm: Spectral clustering with rotation cost heuristic*

Description

Applies spectral clustering to similarity matrix. Number of clusters is based on the rotation cost heuristic.

Usage

```
spectral_rot(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_rot_classic *Clustering algorithm: Spectral clustering with rotation cost heuristic*

Description

Applies spectral clustering to similarity matrix. Number of clusters is based on the rotation cost heuristic.

Usage

```
spectral_rot_classic(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_seven *Clustering algorithm: Spectral clustering for a seven cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks seven clusters.

Usage

```
spectral_seven(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_six *Clustering algorithm: Spectral clustering for a six cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks six clusters.

Usage

```
spectral_six(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_ten *Clustering algorithm: Spectral clustering for a ten cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks ten clusters.

Usage

```
spectral_ten(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_three *Clustering algorithm: Spectral clustering for a three cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks three clusters.

Usage

```
spectral_three(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

spectral_two *Clustering algorithm: Spectral clustering for a two cluster solution*

Description

Applies spectral clustering to similarity matrix. Seeks two clusters.

Usage

```
spectral_two(similarity_matrix)
```

Arguments

similarity_matrix	A similarity matrix.

Value

solution_data A list storing cluster assignments ("solution") and the number of clusters ("nclust").

split_parser *Helper function to determine which row and columns to split on*

Description

Helper function to determine which row and columns to split on

Usage

```
split_parser(
  row_split_vector = NULL,
  column_split_vector = NULL,
  row_split = NULL,
  column_split = NULL,
  n_rows,
  n_columns
)
```

Arguments

row_split_vector	A vector of row indices to split on.
column_split_vector	A vector of column indices to split on.
row_split	Standard parameter of ComplexHeatmap::Heatmap.
column_split	Standard parameter of ComplexHeatmap::Heatmap.
n_rows	The number of rows in the data.
n_columns	The number of columns in the data.

Value

"list"-class object containing row_split and column_split character vectors to pass into ComplexHeatmap::Heatmap.

subc_v *Mock ABCD subcortical volumes data*

Description

Like the mock dataframe "abcd_subc_v", but with "unique_id" as the "uid".

Usage

```
subc_v
```

Format

`subc_v`**:**

A data frame with 174 rows and 31 columns:

unique_id The unique identifier of the ABCD dataset

... Subcortical volumes of various ROIs (mm^3, I think)

Source

Though this data is no longer "real" ABCD data, the reference for using ABCD as a data source is below:

Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive DevelopmentSM (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9-10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators.

`subs` *Select all columns of a dataframe starting with a given string prefix.*

Description

Removes the columns that are not prefixed with 'subject_' prefixed columns from a dataframe. Useful intermediate step for extracting subject UIDs from an solutions_matrix structure.

Usage

```
subs(df)
```

Arguments

`df`	Dataframe

Value

df_subs Dataframe with only 'subject_' prefixed columns

subsample_data_list *Create subsamples of a data_list*

Description

Given a data list, return a list of smaller data lists that are generated through random sampling (without replacement).

Usage

```
subsample_data_list(
  data_list,
  n_subsamples,
  subsample_fraction = NULL,
  n_subjects = NULL
)
```

Arguments

data_list	A nested list of input data from generate_data_list().
n_subsamples	Number of subsamples to create.
subsample_fraction	Percentage of patients to include per subsample.
n_subjects	Number of patients to include per subsample.

Value

A "list" class object containing n_subsamples number of data lists. Each of those data lists contains a random subsample_fraction fraction of the observations of the provided data list.

subsample_pairwise_aris *Calculate pairwise adjusted Rand indices across subsamples of data*

Description

Calculate pairwise adjusted Rand indices across subsamples of data

Usage

```
subsample_pairwise_aris(
  subsample_solutions,
  return_raw_aris = FALSE,
  verbose = FALSE
)
```

Arguments

subsample_solutions	A list of containing cluster solutions from distinct subsamples of the data. This object is generated by the function batch_snf_subsamples().
return_raw_aris	Whether the ARI matrix used to calculate the average ARI across subsamples should be returned.
verbose	If TRUE, print time remaining estimates to console.

Value

If return_raw_aris is FALSE, this function will return

---

summarize_clust_algs_list *Summarize a clust_algs_list object*

Description

Summarize a clust_algs_list object

Usage

```
summarize_clust_algs_list(clust_algs_list)
```

Arguments

clust_algs_list	A clust_algs_list object

Value

summary_df "data.frame" class object containing the name and index of each clustering algorithm in te provided clust_algs_list.

summarize_dl *Summarize a data list*

Description

Summarize a data list

Usage

```
summarize_dl(data_list, scope = "component")
```

Arguments

data_list	A nested list of input data from `generate_data_list()`.
scope	The level of detail for the summary. Options are: • "component" (default): One row per component (dataframe) in the data_list. • "feature": One row for each feature in the data_list.

Value

"data.frame"-class object summarizing all components (or features if scope == "component").

summarize_dml *Summarize metrics contained in a distance_metrics_list*

Description

Summarize metrics contained in a distance_metrics_list

Usage

```
summarize_dml(distance_metrics_list)
```

Arguments

distance_metrics_list	A distance_metrics_list.

Value

"data.frame"-class object summarizing items in a distance metrics list.

summarize_pvals *Summarize p-value columns of an extended solutions matrix*

Description

Summarize p-value columns of an extended solutions matrix

Usage

```
summarize_pvals(extended_solutions_matrix)
```

Arguments

extended_solutions_matrix	Result of `extend_solutions`

Value

The provided extended solutions matrix along with columns for the min, mean, and maximum across p-values for each row.

train_test_assign *Training and testing split*

Description

Given a vector of subject_id and a threshold, returns a list of which members should be in the training set and which should be in the testing set. The function relies on whether or not the absolute value of the Jenkins's one_at_a_time hash function exceeds the maximum possible value (2147483647) multiplied by the threshold.

Usage

```
train_test_assign(train_frac, subjects, seed = 42)
```

Arguments

train_frac	The fraction (0 to 1) of subjects for training
subjects	The available subjects for distribution
seed	Seed used for Jenkins's one_at_a_time hash function

Value

split a named list containing the training and testing subject_ids

`two_step_merge` *Two step SNF*

Description

Individual dataframes into individual similarity matrices into one fused network per domain into one final fused network.

Usage

```
two_step_merge(
  data_list,
  k = 20,
  alpha = 0.5,
  t = 20,
  cont_dist_fn,
  disc_dist_fn,
  ord_dist_fn,
  cat_dist_fn,
  mix_dist_fn,
  weights_row
)
```

Arguments

`data_list`	A nested list of input data from `generate_data_list()`.
`k`	k hyperparameter.
`alpha`	alpha/eta/sigma hyperparameter.
`t`	SNF number of iterations hyperparameter.
`cont_dist_fn`	distance metric function for continuous data.
`disc_dist_fn`	distance metric function for discrete data.
`ord_dist_fn`	distance metric function for ordinal data.
`cat_dist_fn`	distance metric function for categorical data.
`mix_dist_fn`	distance metric function for mixed data.
`weights_row`	dataframe row containing feature weights.

Value

fused_network The final fused network (class "matrix", "array") generated by SNF.

`var_manhattan_plot` *Manhattan plot of feature-feature associaiton p-values*

Description

Manhattan plot of feature-feature associaiton p-values

Usage

```
var_manhattan_plot(
  data_list,
  key_var,
  neg_log_pval_thresh = 5,
  threshold = NULL,
  point_size = 5,
  text_size = 20,
  plot_title = NULL,
  hide_x_labels = FALSE,
  bonferroni_line = FALSE
)
```

Arguments

`data_list`	List of dataframes containing data information.
`key_var`	Feature for which the association p-values of all other features are plotted.
`neg_log_pval_thresh`	Threshold for negative log p-values.
`threshold`	p-value threshold to plot dashed line at.
`point_size`	Size of points in the plot.
`text_size`	Size of text in the plot.
`plot_title`	Title of the plot.
`hide_x_labels`	If TRUE, hides x-axis labels.
`bonferroni_line`	If TRUE, plots a dashed black line at the Bonferroni-corrected equivalent of the p-value threshold.

Value

A Manhattan plot (class "gg", "ggplot") showing the association p-values of features against one key feature in a data list.

Index