Package ‘miRetrieve’

July 22, 2025

Type Package

Title miRNA Text Mining in Abstracts

Version 1.3.4

Author Julian Friedrich [aut, cre],
Hans-Peter Hammes [aut],
Guido Krenning [aut]

Maintainer Julian Friedrich <`julian.friedrich@medma.uni-heidelberg.de`>

Description Providing tools for microRNA (miRNA) text mining. miRetrieve summarizes miRNA literature by extracting, counting, and analyzing miRNA names, thus aiming at gaining biological insights into a large amount of text within a short period of time. To do so, miRetrieve uses regular expressions to extract miRNAs and tokenization to identify meaningful miRNA associations. In addition, miRetrieve uses the latest miRTarBase version 8.0 (Hsi-Yuan Huang et al. (2020) ``miRTarBase 2020: updates to the experimentally validated microRNA–target interaction database" <doi:10.1093/nar/gkz896>) to display field-specific miRNA-mRNA interactions. The most important functions are available as a Shiny web application under <`https://miretrieve.shinyapps.io/miRetrieve/`>.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.1.0)

Imports dplyr (>= 1.0.7), forcats (>= 0.5.1), ggplot2 (>= 3.3.5), magrittr (>= 2.0.1), openxlsx (>= 4.2.4), plotly (>= 4.9.4.1), purrr (>= 0.3.4), readr (>= 2.0.1), readxl (>= 1.3.1), rlang (>= 0.4.11), scales (>= 1.1.1), stringr (>= 1.4.0), textclean (>= 0.9.3), tidyr (>= 1.1.3), tidytext (>= 0.3.1), topicmodels (>= 0.2.12), wordcloud (>= 2.6), xml2 (>= 1.3.2), zoo (>= 1.8-9)

RoxygenNote 7.1.1

Suggests kableExtra, knitr, reshape2, rmarkdown, testthat

NeedsCompilation no

Repository CRAN
Date/Publication 2021-09-18 17:30:02 UTC

Contents

`add_col_topic` *Add topic column to data frame*

Description

Add topic column to a data frame.

Usage

```
add_col_topic(df, col.topic = "Topic", topic.name = "Topic1")
```

Arguments

`df`	Data frame which the topic column is added to.
`col.topic`	String. Name of the topic column to be created.
`topic.name`	String. Topic name to be contained in `col.topic`.

Details

Add a topic column to a data frame. This topic column is named `col.topic` and contains the string `topic.name`.

Value

Data frame with a topic column added.

See Also

`assign_topic()`

`animal_keywords` *Keywords - animals.*

Description

Keywords to identify abstracts using animal models.

Usage

```
animal_keywords
```

Format

An object of class `character` of length 12.

`assign_topic` *Assign topics based on precalculated scores*

Description

Assign topics to abstracts based on precalculated scores.

Usage

```
assign_topic(
  df,
  col.topic,
  threshold,
  topic.names = NULL,
  col.topic.name = "Topic",
  col.pmid = "PMID",
  discard = FALSE
)
```

Arguments

`df`	Data frame containing precalculated topic scores and PubMed-IDs.
`col.topic`	Character vector. Vector with column names containing precalculated topic scores.
`threshold`	Integer vector. Vector containing thresholds for topic columns. Positions in `threshold` correspond to positions in `col.topic`.
`topic.names`	Character vector. Optional. Vector containing names of new topics. Positions in `topic.names` correspond to positions in `col.topic`. If `topic.names` is not provided, `col.topic` is used to name the new topics.
`col.topic.name`	String. Name of the new topic column.
`col.pmid`	String. Column containing PubMed-IDs.
`discard`	Boolean. If `discard = TRUE`, only abstracts with a newly assigned topic are kept. Abstracts without a newly assigned topic are discarded.

Details

Assign topics to abstracts based on precalculated scores. `assign_topic()` compares different precalculated topic scores and assigns the abstract to the topic with the highest score. If there is a tie between topic scores, the abstract is assigned to all topics in question. If an abstract matches no topic, it is assigned to the topic "Unknown".

Value

Data frame with topics based on precalculated topic scores.

See Also

`calculate_score_topic()`, `plot_score_topic()`, `add_col_topic()`

Other score functions: `calculate_score_animals()`, `calculate_score_biomarker()`, `calculate_score_patients()`, `calculate_score_topic()`, `plot_score_animals()`, `plot_score_biomarker()`, `plot_score_patients()`, `plot_score_topic()`

`assign_topic_lda` *Assign topics based on LDA model*

Description

Assign topics to abstracts based on an LDA model.

Usage

```
assign_topic_lda(df, lda_model, topic.names, col.pmid = PMID)
```

Arguments

`df`	Data frame to assign topics to. Should be the same data frame that the LDA model was fitted on.
`lda_model`	LDA-model.
`topic.names`	Character vector. Vector containing names of the new topics. Must have the same length as the number of topics `lda_model` was fitted on.
`col.pmid`	Symbol. Column containing PubMed-IDs.

Details

Assign topic to abstracts based on an LDA model. To identify the subject of a topic, use `plot_lda_term()`.

Value

Data frame with topics assigned to each abstract based on an LDAmodel.

See Also

`fit_lda()`, `plot_lda_term()`, `assign_topic()`

Other LDA functions: `fit_lda()`, `plot_lda_term()`, `plot_perplexity()`

`biomarker_keywords` *Keywords - biomarkers.*

Description

Keywords to identify abstracts reporting about miRNAs as biomarkers.

Usage

```
biomarker_keywords
```

Format

An object of class `character` of length 18.

`calculate_score_animals` *Calculate animal model scores for abstracts*

Description

Calculate animal model score for each abstract to indicate possible use of animal models.

Usage

```
calculate_score_animals(
  df,
  keywords = animal_keywords,
  case = FALSE,
  threshold = NULL,
  indicate = FALSE,
  discard = FALSE,
  col.abstract = Abstract
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. The score is calculated based on these keywords. How much weight a keyword in `keywords` carries is determined by how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score. The predefined keywords can be accessed via `miRetrieve::animal_keywords`.
`case`	Boolean. If `case = TRUE`, terms contained in `keywords` are case sensitive. If `case = FALSE`, terms contained in `keywords` are case insensitive.

`threshold`	Integer. Optional. Threshold to decide if an abstract is considered to use animal models or not. If `indicate = TRUE` or `discard = TRUE` and `threshold` is not specified, `threshold` is automatically set to `1`.
`indicate`	Boolean. If `indicate = TRUE`, an extra column is added. This extra column contains "Yes" or "No", indicating the use of animal models in abstracts.
`discard`	Boolean. If `discard = TRUE`, only abstracts are kept where animal models are present.
`col.abstract`	Symbol. Column containing abstracts.

Details

Calculate animal model score for each abstract to indicate possible use of animal models. This score is added to the data frame as an additional column `Animal_score`, containing the calculated animal model score. To decide which abstracts are considered to contain animal models, a threshold can be set via the `threshold` argument. Furthermore, an additional column can be added, verbally indicating the use of animal models in an abstract. Choosing the right threshold can be facilitated using `plot_score_animals()`.

Value

Data frame with calculated animal model scores. If `discard = FALSE`, adds extra columns to the original data frame with the calculated animal model scores. If `discard = TRUE`, only abstracts with animal models are kept.

See Also

`plot_score_animals()`

Other score functions: `assign_topic()`, `calculate_score_biomarker()`, `calculate_score_patients()`, `calculate_score_topic()`, `plot_score_animals()`, `plot_score_biomarker()`, `plot_score_patients()`, `plot_score_topic()`

---

calculate_score_biomarker

Calculate biomarker scores for abstracts

Description

Calculate biomarker score for each abstract to indicate possible use of miRNAs as biomarker.

Usage

```
calculate_score_biomarker(
  df,
  keywords = biomarker_keywords,
  case = FALSE,
  threshold = NULL,
```

```
  indicate = FALSE,
  discard = FALSE,
  col.abstract = Abstract
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. The score is calculated based on these keywords. How much weight a keyword in `keywords` carries is determined by how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score. The predefined keywords can be accessed via `miRetrieve::biomarker_keywords`.
`case`	Boolean. If `case = TRUE`, terms contained in `keywords` are case sensitive. If `case = FALSE`, terms contained in `keywords` are case insensitive.
`threshold`	Integer. Optional. Threshold to decide if use of miRNAs as biomarker are present in an abstract or not. If `indicate = TRUE` or `discard = TRUE` and `threshold` not specified, `threshold` is automatically set to `1`.
`indicate`	Boolean. If `indicate = TRUE`, an extra column is added. This extra column contains "Yes" or "No", indicating the use of miRNAs as biomarker in abstracts.
`discard`	Boolean. If `TRUE`, only abstracts are kept where miRNAs as biomarker.
`col.abstract`	Symbol. Column containing abstracts.

Details

Calculate biomarker score for each abstract to indicate possible use of miRNAs as biomarker. This score is added to the data frame as an additional column `Biomarker_score`, containing the calculated biomarker score. To decide which abstracts are considered to contain use of miRNAs as biomarker, a threshold can be set via the `threshold` argument. Furthermore, an additional column can be added, verbally indicating the general use of miRNAs as biomarker in an abstract. Choosing the right threshold can be facilitated using `plot_score_biomarker()`.

Value

Data frame with calculated biomarker scores. If `discard = FALSE`, adds extra columns to the original data frame with calculated biomarker scores. If `discard = TRUE`, only abstracts are with miRNAs as biomarker are kept.

See Also

`plot_score_biomarker()`

Other score functions: `assign_topic()`, `calculate_score_animals()`, `calculate_score_patients()`, `calculate_score_topic()`, `plot_score_animals()`, `plot_score_biomarker()`, `plot_score_patients()`, `plot_score_topic()`

---

calculate_score_patients

Calculate patients scores for abstracts

Description

Calculate patients score for each abstract to indicate possible use of patient material.

Usage

```
calculate_score_patients(
  df,
  keywords = patients_keywords,
  case = FALSE,
  threshold = NULL,
  indicate = FALSE,
  discard = FALSE,
  col.abstract = Abstract
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. The score is calculated based on these keywords. How much weight a keyword in `keywords` carries is determined by how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score. The predefined keywords can be accessed via `miRetrieve::patients_keywords`.
`case`	Boolean. If `case = TRUE`, terms contained in `keywords` are case sensitive. If `case = FALSE`, terms contained in `keywords` are case insensitive.
`threshold`	Integer. Optional. Threshold to decide if use of patient tissue is present in an abstract or not. If `indicate = TRUE` or `discard = TRUE` and `threshold` not specified, `threshold` is automatically set to `1`.
`indicate`	Boolean. If `indicate = TRUE`, an extra column is added. This extra column contains "Yes" or "No", indicating the use of patient tissue in abstracts.
`discard`	Boolean. If `discard = TRUE`, only abstracts are kept where use of patient tissue is present.
`col.abstract`	Symbol. Column containing abstracts.

Details

Calculate patient score for each abstract to indicate possible use of patient material. This score is added to the data frame as an additional column `Patient_score`, containing the calculated patients score. To decide which abstracts are considered to contain patient material, a threshold can be set via the `threshold` argument. Furthermore, an additional column can be added, verbally indicating the general use of patient material. Choosing the right threshold can be facilitated using `plot_score_patients()`.

Value

Data frame with calculated patient scores. If `discard` = `FALSE`, adds extra columns to the original data frame with the calculated patient tissue scores. If `discard` = `TRUE`, only abstracts with use of patient tissue are kept.

See Also

`plot_score_patients()`

Other score functions: `assign_topic()`, `calculate_score_animals()`, `calculate_score_biomarker()`, `calculate_score_topic()`, `plot_score_animals()`, `plot_score_biomarker()`, `plot_score_patients()`, `plot_score_topic()`

`calculate_score_topic` *Calculate scores of a self-chosen topic*

Description

Calculate score of a self-chosen topic for each abstract to identify abstracts possibly corresponding to the topic of interest.

Usage

```
calculate_score_topic(
  df,
  keywords,
  case = FALSE,
  col.score = "topic_score",
  col.indicate = NULL,
  threshold = NULL,
  discard = FALSE,
  col.abstract = Abstract
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. The score is calculated based on these keywords. How much weight a keyword in `keywords` carries is determined by how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score.
`case`	Boolean. If `case` = `TRUE`, terms contained in `keywords` are case sensitive. If `case` = `FALSE`, terms contained in `keywords` are case insensitive.
`col.score`	String. Name of `topic_score` column.

`col.indicate`	String. Optional. Name of indicating column. If a string is provided, an extra column is added to `df`, indicating if the abstract corresponds to the topic of interest by "Yes" or "No".
`threshold`	Integer. Optional. Threshold to decide if abstract corresponds to topic of interest. If `col.topic` is specified or `discard = TRUE` without `threshold` being specified, `threshold` is automatically set to `1`.
`discard`	Boolean. If `discard = TRUE`, only abstracts are kept that correspond to the topic of interest.
`col.abstract`	Symbol. Column containing abstracts.

Details

Calculate score of a self-chosen topic for each abstract to identify abstracts possibly corresponding to the topic of interest. This score is added to the data frame as an additional column, usually called `topic_score`, containing the calculated topic score. If there is more than one topic of interest, the column `topic_score` should be appropriately renamed. To decide which abstracts are considered to correspond to the topic of interest, a threshold can be set via the `threshold` argument. Furthermore, an additional column can be added, verbally indicating if the abstract corresponds to the topic. Choosing the right threshold can be facilitated using `plot_score_topic()`.

Value

Data frame with calculated topic scores. If `discard = FALSE`, adds extra columns to the original data frame with the calculated topic scores. If `discard = TRUE`, only abstracts corresponding to the topic of interest are kept.

See Also

`assign_topic()`, `plot_score_topic()`

Other score functions: `assign_topic()`, `calculate_score_animals()`, `calculate_score_biomarker()`, `calculate_score_patients()`, `plot_score_animals()`, `plot_score_biomarker()`, `plot_score_patients()`, `plot_score_topic()`

`combine_df` *Combine data frames into one data frame*

Description

Combine data frames into one data frame.

Usage

```
combine_df(...)
```

Arguments

`...`	Data frames to combine into one data frame. Data frames must have the same number of columns and the same column names.

Details

Combine data frames into one data frame. `combine_df()` accepts several data frames that are combined into one data frame. Data frames to be combined must have the same number of columns and the same column names.

Value

Combined data frame.

See Also

Other combine functions: `combine_mir()`

`combine_mir` *Combine miRNA vectors into one*

Description

Combine miRNA vectors into one.

Usage

```
combine_mir(...)
```

Arguments

`...`	Character vectors. Character vectors containing miRNA names.

Details

Combine miRNA vectors into one. miRNA names occurring more than once are reduced to one instance.

Value

Combined character vector containing miRNA names.

See Also

`get_mir()`

Other combine functions: `combine_df()`

`combine_stopwords` *Combine data frames containing stop words*

Description

Combine data frames containing stop words into one data frame.

Usage

```
combine_stopwords(...)
```

Arguments

`...`	Data frames with stop words. Data frames must have two columns named "word" and "lexicon".

Details

Combine data frames containing stop words into one data frame. Provided data frames must have two columns named "word" and "lexicon".

Value

Combined data frame with stop words.

See Also

`generate_stopwords()`, stopwords_miretrieve, tidytext::stop_words

Other stopword functions: `generate_stopwords()`

`compare_mir_count` *Compare count of miRNA names between different topics*

Description

Compare count of miRNA names between different topics.

Usage

```
compare_mir_count(
  df,
  mir,
  topic = NULL,
  normalize = TRUE,
  col.topic = Topic,
  col.mir = miRNA,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing columns for miRNA names, topics, and PubMed-IDs.
`mir`	Character vector. Vector specifying which miRNA names to compare.
`topic`	Character vector. Optional. Vector specifying which topics to compare.
`normalize`	Boolean. If `normalize = TRUE`, plot the proportion of abstracts mentioning a miRNA name compared to all abstracts in a topic. If `normalize = FALSE`, plot the absolute number of abstracts mentioning a miRNA in a topic.
`col.topic`	Symbol. Column containing topic names.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Compare count of miRNA names between different topics by plotting the number of abstracts mentioning the miRNA in a topic. This count can either be normalized, thus plotting the proportion of abstracts mentioning a miRNA name compared to all abstracts of a topic, or it can be not normalized, thus plotting the absolute number of abstracts mentioning a miRNA per topic.

Value

Bar plot comparing the count of miRNA names between different topics.

See Also

`compare_mir_count_log2()`, `compare_mir_count_unique()`

Other compare functions: `compare_mir_count_log2()`, `compare_mir_count_unique()`, `compare_mir_terms_log2()`, `compare_mir_terms_scatter()`, `compare_mir_terms_unique()`, `compare_mir_terms()`

`compare_mir_count_log2` *Compare log2-frequency count of miRNA names between two topics*

Description

Compare log2-frequency count of miRNA names between two topics

Usage

```
compare_mir_count_log2(
  df,
  mir,
  topic = NULL,
  normalize = TRUE,
  col.topic = Topic,
  col.mir = miRNA,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names, topics, and PubMed-IDs.
`mir`	Character vector. Vector specifying which miRNA names to compare.
`topic`	Character vector. Optional. Vector specifying which topics to compare. If `topic = NULL`, all topics in `df` are used.
`normalize`	Boolean. If `normalize = TRUE`, proportion of abstracts mentioning a miRNA name compared to all abstracts of a topic are used. If `normalize = FALSE`, the absolute number of abstracts mentioning a miRNA name is used.
`col.topic`	Symbol. Column containing topics.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Compare log2-frequency count of miRNA names between two topics by plotting the log2-ratio of the miRNA count in two topics. The miRNA count per topic can either be normalized, thus taking the proportion of abstracts mentioning a miRNA name compared to all abstracts in a topic, or not normalized, thus taking the absolute number of abstracts mentioning a miRNA in a topic. The log2-plot is greatly inspired by the book “tidytext: Text Mining and Analysis Using Tidy Data Principles in R.” by Silge and Robinson.

Value

List containing bar plot comparing the log2-frequency count of miRNA names between two topics and its corresponding data frame.

References

Silge, Julia, and David Robinson. 2016. “tidytext: Text Mining and Analysis Using Tidy Data Principles in R.” JOSS 1 (3). The Open Journal. https://doi.org/10.21105/joss.00037.

See Also

`compare_mir_count()`, `compare_mir_count_unique()`

Other compare functions: `compare_mir_count_unique()`, `compare_mir_count()`, `compare_mir_terms_log2()`, `compare_mir_terms_scatter()`, `compare_mir_terms_unique()`, `compare_mir_terms()`

`compare_mir_count_unique` *Compare top count of unique miRNA names per topic*

Description

Compare top count of unique miRNA names per topic

Usage

```
compare_mir_count_unique(
  df,
  top = 5,
  topic = NULL,
  normalize = TRUE,
  colour = "steelblue3",
  col.topic = Topic,
  col.mir = miRNA,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names, topics, and PubMed-IDs.
`top`	Integer. Specifies number of top unique miRNAs to plot.
`topic`	Character vector. Optional. Vector specifying which topics to compare. If `topic = NULL`, all topics in `df` are used.
`normalize`	Boolean. If `normalize = TRUE`, proportion of abstracts mentioning a miRNA name compared to all abstracts of a topic are used. If `normalize = FALSE`, the absolute number of abstracts mentioning a miRNA name is used.

`colour`	String. Colour of bar plot.
`col.topic`	Symbol. Column containing topics.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Compare top count of unique miRNA names per topic by plotting the the miRNA count of unique miRNAs per topic. Per topic, the unique miRNAs are identified and their count is plotted. The miRNA count can either be normalized, thus taking the proportion of abstracts mentioning a miRNA name compared to all abstracts in a topic, or not normalized, thus taking the absolute number of abstracts mentioning a miRNA in a topic.

Value

Bar plot comparing frequency of unique miRNA count per topic.

See Also

`compare_mir_count()`, `compare_mir_count_log2()`

Other compare functions: `compare_mir_count_log2()`, `compare_mir_count()`, `compare_mir_terms_log2()`, `compare_mir_terms_scatter()`, `compare_mir_terms_unique()`, `compare_mir_terms()`

`compare_mir_terms` *Compare count of terms associated with a miRNA name over various topics*

Description

Compare count of top terms associated with a miRNA name over various topics.

Usage

```
compare_mir_terms(
  df,
  mir,
  top = 20,
  token = "words",
  ...,
  topic = NULL,
  shared = TRUE,
  normalize = TRUE,
  stopwords = stopwords_miretrieve,
  stopwords_ngram = TRUE,
  position = "dodge",
```

```
  col.mir = miRNA,
  col.abstract = Abstract,
  col.topic = Topic,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names, abstracts, topics, and PubMed-IDs.
`mir`	String. miRNA name of interest.
`top`	Integer. Number of top terms to plot.
`token`	String. Specifies how abstracts shall be split up. Taken from `unnest_tokens()` in the **tidytext** package: "Unit for tokenizing, or a custom tokenizing function. Built-in options are "words" (default), "characters", "character_shingles", "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", "regex", (...), and "ptb" (Penn Treebank). If a function, should take a character vector and return a list of character vectors of the same length."
`...`	Additional arguments for tokenization, if necessary.
`topic`	Character vector. Optional. Specifies topics to plot. If `topic = NULL`, all topics in `df` are plotted.
`shared`	Boolean. If `shared = TRUE`, only terms that are shared between all topics are plotted.
`normalize`	Boolean. If `normalize = TRUE`, normalizes the number of abstracts to the total number of abstracts with a miRNA name in a topic.
`stopwords`	Data frame containing stop words.
`stopwords_ngram`	Boolean. Specifies if stop words shall be removed from abstracts when using ngrams. Only applied when `token = 'ngrams'`.
`position`	Character vector. Vector containing either "dodge" or "facet". Determines if bar plots are on top of or next to each other.
`col.mir`	Symbol. Column containing miRNA names.
`col.abstract`	Symbol. Column containing abstracts.
`col.topic`	Symbol. Column containing topic names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Compare count of top terms associated with a miRNA name over various topics. miRNA names and topics must be in a data frame `df`, while terms are taken from abstracts contained in `df`. Number of top terms to plot is regulated by `top`. Terms can either be evaluated as their raw count, e.g. in how many abstracts they are mentioned in conjunction with the miRNA name, or as their relative count, e.g. in how many abstracts containing the miRNA they are mentioned compared to all abstracts containing the miRNA. `compare_mir_terms()` is based on the tools available in the **tidytext** package.

Value

Bar plot comparing the count of terms associated with a miRNA name over two topics.

See Also

`compare_mir_terms_log2()`, `compare_mir_terms_scatter()`

Other compare functions: `compare_mir_count_log2()`, `compare_mir_count_unique()`, `compare_mir_count()`, `compare_mir_terms_log2()`, `compare_mir_terms_scatter()`, `compare_mir_terms_unique()`

`compare_mir_terms_log2` *Compare log2-frequency count of terms associated with a miRNA name*

Description

Compare log2-frequency count of terms associated with a miRNA name over two topics.

Usage

```
compare_mir_terms_log2(
  df,
  mir,
  top = 20,
  token = "words",
  ...,
  topic = NULL,
  shared = TRUE,
  normalize = TRUE,
  stopwords = stopwords_miretrieve,
  stopwords_ngram = TRUE,
  col.mir = miRNA,
  col.abstract = Abstract,
  col.topic = Topic,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names, abstracts, topics, and PubMed-IDs.
`mir`	String. miRNA name of interest.
`top`	Integer. Number of top terms to plot.

`token`	String. Specifies how abstracts shall be split up. Taken from `unnest_tokens()` in the **tidytext** package: "Unit for tokenizing, or a custom tokenizing function. Built-in options are "words" (default), "characters", "character_shingles", "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", "regex", (...), and "ptb" (Penn Treebank). If a function, should take a character vector and return a list of character vectors of the same length."
`...`	Additional arguments for tokenization, if necessary.
`topic`	Character vector. Optional. Specifies which topics to plot. Must have length two. If `topic = NULL`, all topics in `df` are plotted.
`shared`	Boolean. If `shared = TRUE`, only terms that are shared between the two topics are plotted.
`normalize`	Boolean. If `normalize = TRUE`, normalizes the number of abstracts to the total number of abstracts in a topic.
`stopwords`	Data frame containing stop words.
`stopwords_ngram`	Boolean. Specifies if stop words shall be removed from abstracts when using ngrams. Only applied when `token = 'ngrams'`.
`col.mir`	Symbol. Column containing miRNA names.
`col.abstract`	Symbol. Column containing abstracts.
`col.topic`	Symbol. Column containing topic names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Compare log2-frequency count of terms associated with a miRNA name over two topics by plotting the log2-ratio of the term count associated with a miRNA name over two topics. miRNA names and topics must be in a data frame `df`, while terms are taken from abstracts contained in `df`. Number of top terms to plot is regulated by `top`. Terms can either be evaluated as their raw count, e.g. in how many abstracts they are mentioned in conjunction with the miRNA name, or as their relative count, e.g. in how many abstracts containing the miRNA they are mentioned compared to all abstracts containing the miRNA. `compare_mir_terms_log2()` is based on the tools available in the **tidytext** package. The log2-plot is greatly inspired by the book “tidytext: Text Mining and Analysis Using Tidy Data Principles in R.” by Silge and Robinson.

Value

List containing bar plot comparing the log2-frequency of terms associated with a miRNA over two topics and its corresponding data frame.

References

Silge, Julia, and David Robinson. 2016. “tidytext: Text Mining and Analysis Using Tidy Data Principles in R.” JOSS 1 (3). The Open Journal. https://doi.org/10.21105/joss.00037.

See Also

`compare_mir_terms()`, `compare_mir_terms_scatter()`

Other compare functions: `compare_mir_count_log2()`, `compare_mir_count_unique()`, `compare_mir_count()`, `compare_mir_terms_scatter()`, `compare_mir_terms_unique()`, `compare_mir_terms()`

---

compare_mir_terms_scatter

Compare shared terms associated with a miRNA name

Description

Compare shared terms associated with a miRNA name over two topics.

Usage

```
compare_mir_terms_scatter(
  df,
  mir,
  top = 1000,
  token = "words",
  ...,
  topic = NULL,
  stopwords = stopwords_miretrieve,
  stopwords_ngram = TRUE,
  html = TRUE,
  colour.point = "red",
  colour.term = "black",
  col.mir = miRNA,
  col.abstract = Abstract,
  col.topic = Topic,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names, abstracts, topics, and PubMed-IDs.
`mir`	String. miRNA name of interest.
`top`	Integer. Number of top terms to plot.
`token`	String. Specifies how abstracts shall be split up. Taken from `unnest_tokens()` in the **tidytext** package: "Unit for tokenizing, or a custom tokenizing function. Built-in options are "words" (default), "characters", "character_shingles", "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", "regex", (...), and "ptb" (Penn Treebank). If a function, should take a character vector and return a list of character vectors of the same length."

`...`	Additional arguments for tokenization, if necessary.
`topic`	Character vector. Optional. Specifies which topics to plot. Must have length two. If `topic = NULL`, all topics in `df` are plotted.
`stopwords`	Data frame containing stop words.
`stopwords_ngram`	Boolean. Specifies if stop words shall be removed from abstracts when using ngrams. Only applied when `token = 'ngrams'`.
`html`	Boolean. Specifies if plot is returned as an HTML-widget or static.
`colour.point`	String. Colour of points for scatter plot.
`colour.term`	String. Colour of terms for scatter plot.
`col.mir`	Symbol. Column containing miRNAs.
`col.abstract`	Symbol. Column containing abstracts.
`col.topic`	Symbol. Column containing topics names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Compare shared terms associated with a miRNA name over two topics. These terms are displayed as a scatter plot, which is either interactive as an HTML-widget, or static. This is regulated via the `html` argument. miRNA names and topics must be in a data frame `df`, while terms are taken from abstracts contained in `df`. Number of top terms to choose is regulated by `top`. Terms are evaluated as their raw count and plotted on a log10-scale. `compare_mir_terms_scatter()` is based on the tools available in the **tidytext** package. The term-plot is greatly inspired by “tidytext: Text Mining and Analysis Using Tidy Data Principles in R.” by Silge and Robinson.

Value

Scatter plot comparing shared terms of a miRNA between two topics.

References

Silge, Julia, and David Robinson. 2016. “tidytext: Text Mining and Analysis Using Tidy Data Principles in R.” JOSS 1 (3). The Open Journal. https://doi.org/10.21105/joss.00037.

See Also

`compare_mir_terms()`, `compare_mir_terms_log2()`

Other compare functions: `compare_mir_count_log2()`, `compare_mir_count_unique()`, `compare_mir_count()`, `compare_mir_terms_log2()`, `compare_mir_terms_unique()`, `compare_mir_terms()`

`compare_mir_terms_unique`

Compare terms uniquely associated with a miRNA name

Description

Compare terms uniquely associated with a miRNA name over topics.

Usage

```
compare_mir_terms_unique(
  df,
  mir,
  top = 20,
  token = "words",
  ...,
  topic = NULL,
  stopwords = stopwords_miretrieve,
  stopwords_ngram = TRUE,
  normalize = TRUE,
  colour = "steelblue3",
  col.mir = miRNA,
  col.abstract = Abstract,
  col.topic = Topic,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names, abstracts, topics, and PubMed-IDs.
`mir`	String. miRNA name of interest.
`top`	Integer. Number of top terms to plot.
`token`	String. Specifies how abstracts shall be split up. Taken from `unnest_tokens()` in the **tidytext** package: "Unit for tokenizing, or a custom tokenizing function. Built-in options are "words" (default), "characters", "character_shingles", "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", "regex", (...), and "ptb" (Penn Treebank). If a function, should take a character vector and return a list of character vectors of the same length."
`...`	Additional arguments for tokenization, if necessary.
`topic`	Character vector. Optional. Specifies which topics to plot. If `topic = NULL`, all topics in `df` are plotted.
`stopwords`	Data frame containing stop words.
`stopwords_ngram`	Boolean. Specifies if stop words shall be removed from abstracts when using ngrams. Only applied when `token = 'ngrams'`.

`normalize`	Boolean. If `normalize = TRUE`, relative term frequency is plotted, denoting the relative number of papers with `mir` mentioning the term compared to all papers with `mir` mentioning the term. If `normalize = FALSE`, absolute term frequency is plotted, denoting the number of papers with `mir` the term is mentioned in.
`colour`	String. Colour of bar plot.
`col.mir`	Symbol. Column containing miRNAs.
`col.abstract`	Symbol. Column containing abstracts.
`col.topic`	Symbol. Column containing topics names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Compare terms uniquely associated with a miRNA name over topics. miRNA names and topics must be in a data frame `df`, while terms are taken from abstracts contained in `df`. Number of top terms to choose is regulated by `top`. Terms are evaluated either as the number of times they are mentioned in all abstracts with the miRNA name of interest, or the number of times they are relatively mentioned compared to all abstracts with the miRNA name of interest. `compare_mir_terms_unique()` is based on the tools available in the **tidytext** package.

Value

Bar plot containing unique miRNA-terms associations per topic.

See Also

`compare_mir_terms()`, `compare_mir_terms_log2()`, `compare_mir_terms_scatter()`

Other compare functions: `compare_mir_count_log2()`, `compare_mir_count_unique()`, `compare_mir_count()`, `compare_mir_terms_log2()`, `compare_mir_terms_scatter()`, `compare_mir_terms()`

`count_mir` *Count miRNA names in a data frame*

Description

Count occurrence of miRNA names in a data frame.

Usage

```
count_mir(df, col.mir = miRNA)
```

Arguments

`df`	Data frame containing miRNA names.
`col.mir`	Symbol. Column containing miRNA names.

Details

Count occurrence of miRNA names in a data frame. The count of miRNA names is returned as a separate data frame, only listing the miRNA names and their respective frequency.

Value

Data frame. Data frame containing miRNA names and their respective frequency.

See Also

`plot_mir_count()`, `count_mir_threshold()`, `plot_mir_count_threshold()`

Other count functions: `count_mir_threshold()`, `count_snp()`, `plot_mir_count_threshold()`, `plot_mir_count()`

`count_mir_threshold` *Count occurrence of miRNA names above threshold*

Description

Count occurrence of miRNA names above a threshold.

Usage

```
count_mir_threshold(df, threshold = 1, col.mir = miRNA, col.pmid = PMID)
```

Arguments

`df`	Data frame containing miRNA names and PubMed-IDs.
`threshold`	Integer or float. If `threshold` >= 1, counts number of miRNA names in at least `threshold` abstracts. If `threshold` is between 0 and 1, counts number of miRNA names mentioned in at least `threshold` abstracts of all abstracts in `df`.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.

Details

Count occurrence of miRNA names above a threshold. This threshold can either be an absolute value, e.g. 3, or a float between 0 and 1, e.g. 0.2. If `threshold` is an absolute value, number of distinct miRNA names mentioned in at least `threshold` abstracts is returned. If `threshold` is a float between 0 and 1, number of distinct miRNA names mentioned in at least `threshold` abstracts of all abstracts in `df` is returned.

Value

Integer with the number of distinct miRNA names in `df`.

See Also

`plot_mir_count_threshold()`, `count_mir()`, `plot_mir_count()`

Other count functions: `count_mir()`, `count_snp()`, `plot_mir_count_threshold()`, `plot_mir_count()`

`count_snp` *Count SNPs in a data frame*

Description

Count occurrence of SNPs in a data frame.

Usage

```
count_snp(df, col.snp = SNPs, col.pmid = PMID)
```

Arguments

`df`	Data frame containing SNPs and PubMed IDs.
`col.snp`	Symbol. Column containing SNPs.
`col.pmid`	Symbol. Column containing PubMed IDs.

Details

Count occurrence of SNPs in a data frame. The count of SNPs is returned as a separate data frame, only listing the SNPs and their respective frequency.

Value

Data frame. Data frame containing SNPs and their respective frequency.

See Also

`extract_snp()`, `get_snp()`, `subset_snp()`

Other count functions: `count_mir_threshold()`, `count_mir()`, `plot_mir_count_threshold()`, `plot_mir_count()`

`count_target` *Count targets in data frame*

Description

Count occurrence of targets in a data frame.

Usage

```
count_target(df, col.target = Target, add.df = TRUE)
```

Arguments

`df`	Data frame containing a column with targets.
`col.target`	Symbol. Column containing targets.
`add.df`	Boolean. If `add.df` = `TRUE`, adds column `Target_count` to `df` containing the count of targets. If `add.df` = `FALSE`, returns a new data frame with the count of targets.

Details

Count occurrence of targets in a data frame. The count of targets can either be returned as a separate data frame, only listing the targets and their respective frequency, or it can be added to the data frame provided as an extra column.

Value

Data frame, either with the targets and their frequency as a new data frame, or with the frequency of targets added as a new column to the input data frame `df`.

See Also

`join_targets()`, `plot_target_count()`

Other target functions: `join_mirtarbase()`, `join_targets()`, `plot_target_count()`, `plot_target_mir_scatter()`

`df_crc` *Dataset of PubMed data of miRNAs in Colorectal Cancer*

Description

A dataset PubMed abstracts of miRNAs in Colorectal Cancer.

Usage

```
df_crc
```

Format

A data frame.

Source

https://pubmed.ncbi.nlm.nih.gov/

`df_mirtarbase` *miRTarBase version 8.0*

Description

The most recent miRTarBase version 8.0, containing miRNA stem, capitalized targets, and PMIDs.

Usage

`df_mirtarbase`

Format

A data frame with the columns "miRNA_tarbase", "Target", and "PMID".

Details

miRTarBase was published in

Hsi-Yuan Huang, Yang-Chi-Dung Lin, Jing Li, et al., miRTarBase 2020: updates to the experimentally validated microRNA–target interaction database, Nucleic Acids Research, Volume 48, Issue D1, 08 January 2020, Pages D148–D154, https://doi.org/10.1093/nar/gkz896

Source

https://miRTarBase.cuhk.edu.cn:443/

`df_panc` *Dataset of PubMed data of miRNAs in Pancreatic Cancer*

Description

A dataset PubMed abstracts of miRNAs in Pancreatic Cancer.

Usage

`df_panc`

Format

A data frame.

Source

https://pubmed.ncbi.nlm.nih.gov/

`df_test` *Test dataset of PubMed abstracts*

Description

Test dataset of 20 PubMed abstracts.

Usage

```
df_test
```

Format

A data frame.

Source

https://pubmed.ncbi.nlm.nih.gov/

`extract_mir_df` *Extract miRNA names from abstracts in data frame*

Description

Extract miRNA names from abstracts in a data frame.

Usage

```
extract_mir_df(
  df,
  threshold = 1,
  col.abstract = Abstract,
  extract_letters = FALSE
)
```

Arguments

`df`	Data frame containing abstracts.
`threshold`	Integer. Specifies how often a miRNA must be mentioned in an abstract to be extracted.
`col.abstract`	Symbol. Column containing abstracts.
`extract_letters`	Boolean. If `extract_letters = FALSE`, only the miRNA stem is extracted (e.g. *miR-23*). If `extract_letters = TRUE`, the miRNA stem with trailing letter (e.g. *miR-23a*) is extracted.

Details

Extract miRNA names from abstracts in a data frame. miRNA names can either be extracted with their stem only, e.g. *miR-23*, or with their trailing letter, e.g. *miR-23a*. miRNA names are adapted to the most recent miRBase version (e.g. miR-97, miR-102, miR-180(a/b) become miR-30a, miR-29a, and miR-172(a/b), respectively). Additionally, how often a miRNA must be mentioned in an abstract to be extracted can be regulated via the `threshold` argument. Ultimately, abstracts not containing any miRNA names are silently dropped. As many abstracts do not adhere to the miRNA nomenclature, it is recommended to extract only the miRNA stem with `extract_letters = FALSE`.

Value

Data frame with miRNA names extracted from abstracts.

See Also

`extract_mir_string()`

Other extract functions: `extract_mir_string()`, `extract_snp()`

`extract_mir_string` *Extract miRNA names from string*

Description

Extract miRNA names from a string.

Usage

```
extract_mir_string(string, threshold = 1, extract_letters = FALSE)
```

Arguments

`string`	String. String to search for miRNA names.
`threshold`	Integer. Specifies how often a miRNA must be mentioned in `string` to be extracted.
`extract_letters`	Boolean. If `extract_letters = FALSE`, only the miRNA stem is extracted (e.g. *miR-23*). If `extract_letters = TRUE`, the miRNA stem with trailing letter (e.g. *miR-23a*) is extracted.

Details

Extract miRNA names from a string. miRNA names can either be extracted with their stem only, e.g. *miR-23*, or with their trailing letter, e.g. *miR-23a*. Furthermore, miRNA names are adapted to the most recent miRBase version (e.g. miR-97, miR-102, miR-180(a/b) become miR-30a, miR-29a, and miR-172(a/b), respectively).

Value

Character vector containing miRNA names, if miRNA names are present in the string. If no miRNA names are present in the string, a message is returned saying *"No miRNA found."*.

See Also

`extract_mir_df()`

Other extract functions: `extract_mir_df()`, `extract_snp()`

`extract_snp` *Extract SNPs from abstracts in data frame*

Description

Extract SNPs from abstracts in a data frame.

Usage

```
extract_snp(
  df,
  pattern = snp_pattern,
  col.abstract = Abstract,
  indicate = FALSE,
  discard = FALSE
)
```

Arguments

`df`	Data frame containing abstracts.
`pattern`	String. Regex pattern to identify SNPs.
`col.abstract`	Symbol. Column containing abstracts.
`indicate`	Boolean. If `indicate = TRUE`, add another column called "SNP_present", verbally indicating if a SNP is present in an abstract.
`discard`	Boolean. If `discard = TRUE`, only abstracts containing a SNP are kept.

Details

Extract SNPs from abstracts in a data frame. SNPs are added to the data frame in a separate column. Furthermore, an optional column can indicate if SNPs are generally present in an abstract.

Value

Data frame. If `discard = FALSE`, return the data frame with an additional column for SNPs. If `discard = TRUE`, return only abstracts containing SNPs.

See Also

`count_snp()`, `get_snp()`, `subset_snp()`

Other extract functions: `extract_mir_df()`, `extract_mir_string()`

`fit_lda` *Fit LDA-model*

Description

Fit LDA-model with k topics.

Usage

```
fit_lda(
  df,
  k,
  stopwords = stopwords_miretrieve,
  method = "gibbs",
  control = NULL,
  seed = 42,
  col.abstract = Abstract,
  col.pmid = PMID
)
```

Arguments

`df`	Data frame containing abstracts and PubMed-IDs.
`k`	Integer. Number of topics to fit. Must be >=2.
`stopwords`	Data frame containing stop words.
`method`	String. Either `"gibbs"` or `"VEM"`.
`control`	Control parameters for LDA modeling. For more information, see the documentation of the `LDAcontrol` class in the **topicmodels** package.
`seed`	Integer. Seed for reproducibility.
`col.abstract`	Column containing abstracts.
`col.pmid`	Column containing PubMed-ID.

Details

Fit LDA-model with `k` topics from a data frame. `fit_lda()` is based on `LDA()` from the package **topicmodels**.

Value

LDA-model.

See Also

`plot_perplexity()`

Other LDA functions: `assign_topic_lda()`, `plot_lda_term()`, `plot_perplexity()`

`generate_stopwords` *Generate data frame containing stop words*

Description

Generate a data frame containing stop words.

Usage

```
generate_stopwords(stopwords, combine_with = NULL)
```

Arguments

`stopwords`	Character vector. Vector containing stop words.
`combine_with`	Data frame containing stop words. Optional. Data frame provided here must have only two columns, namely `word` and `lexicon`. This data frame is combined with the data frame created from `stopwords`. Exemplary data frames are • `tidytext::stop_words` from the **tidytext** package, or • `stopwords_miretrieve` from this package.

Details

Generate data frame containing stop words from a character vector. This data frame consists of two columns, namely `word`, containing the stop words, and `lexicon`, containing the string "self-defined". Additionally, the created data frame can be combined with other stop words containing data frames, e.g. `tidytext::stop_words` or `stopwords_miretrieve`.

Value

Data frame containing stop words.

References

Silge, Julia, and David Robinson. 2016. "tidytext: Text Mining and Analysis Using Tidy Data Principles in R." JOSS 1 (3). The Open Journal. https://doi.org/10.21105/joss.00037.

See Also

`combine_stopwords()`, stopwords_miretrieve, tidytext::stop_words

Other stopword functions: `combine_stopwords()`

`get_distinct_mir_df` *Identify top miRNA names distinct for one topic compared to another topic*

Description

Identify top miRNA names distinct for one topic compared to another topic in a data frame.

Usage

```
get_distinct_mir_df(
  df,
  distinct,
  top = 5,
  topic = NULL,
  col.topic = Topic,
  col.mir = miRNA,
  col.pmid = PMID
)
```

Arguments

`df`	Data frame containing at least two topics and miRNA names.
`distinct`	String. Name of topic top distinct miRNAs shall be identified for. `distinct` must be contained in the topic names provided in `topic`.
`top`	Integer. Number of top miRNA names to extract for both topics.

`topic`	String. Vector of strings containing topic names to compare miRNA names for. If `topic` = `NULL`, `topic` defaults to all topic names contained in `col.topic` in `df`. `topic` must only contain two topic names.
`col.topic`	Symbol. Column containing topic names.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.

Details

Get top distinct miRNA names of one topic compared to another topic in a data frame. `get_distinct_mir_df()` compares the top miRNA names of two topics and returns the miRNA names that are exclusive for `distinct`.

Value

Character vector containing miRNA names distinct for `distinct` compared to the second topic provided in `topic`.

See Also

Other get functions: `get_distinct_mir_vec()`, `get_mir()`, `get_pmid()`, `get_shared_mir_df()`, `get_shared_mir_vec()`, `get_snp()`

`get_distinct_mir_vec` *Identify miRNA names distinct for one vector compared to another vector*

Description

Identify miRNA names distinct for one vector compared to another vector.

Usage

```
get_distinct_mir_vec(mirna.vec.1, mirna.vec.2)
```

Arguments

`mirna.vec.1`	Character vector. First vector containing miRNA names.
`mirna.vec.2`	Character vector. Second vector containing miRNA names.

Details

Get distinct miRNA names of one vector compared to another vector. `get_distinct_mir()` compares two vectors containing miRNA names and returns the miRNA names that are exclusive for `mirna.vec.1`.

Value

Character vector containing miRNA names distinct for `mirna.vec.1` compared to `mirna.vec.2`.

See Also

Other get functions: `get_distinct_mir_df()`, `get_mir()`, `get_pmid()`, `get_shared_mir_df()`, `get_shared_mir_vec()`, `get_snp()`

`get_mir` *Get miRNA names from a data frame*

Description

Get miRNA names from a data frame. These miRNA names can either be the most frequent ones, or the ones exceeding a threshold.

Usage

```
get_mir(
  df,
  top = NULL,
  threshold = NULL,
  topic = NULL,
  col.mir = miRNA,
  col.pmid = PMID,
  col.topic = Topic
)
```

Arguments

`df`	Data frame containing miRNA names. If `threshold` is set, `df` must also contain PubMed-IDs. If `topic` is set, `df` must also contain topic names.
`top`	Integer. Optional. Specifies number of most frequent miRNA names to return. If neither `top` nor `threshold` is set, `top` is automatically set to `5`.
`threshold`	Integer or float. Optional. If `threshold` >= 1, return miRNA names mentioned in at least `threshold` abstracts. If `threshold` is between 0 and 1, return miRNA names mentioned in at least `threshold` abstracts of all abstracts in `df`.
`topic`	String. Optional. Character vector specifying which topics to obtain miRNA names from.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`col.topic`	Symbol. Column containing topic names.

Details

Get miRNA names from a data frame. These miRNA names can either be the most frequent ones, or the ones exceeding a threshold. Furthermore, if the data frame contains abstracts of different topics, only the miRNA names of specific topics can be obtained by setting the `topic` argument.

- To get the most frequent miRNA names, set the `top` argument. `top` determines how many most frequent miRNA names are returned, according to their rank. Ties among the most frequently mentioned miRNAs are treated as the same rank, e.g. if *miR-126*, *miR-34*, and *miR-29* were all mentioned the most often with the same frequency, they would all be returned by specifying `top = 1`, `top = 2`, and `top = 3`.
- To get the miRNA names exceeding a threshold, set the `threshold` argument. `threshold` can either be an absolute value, e.g. 3, or a float between 0 and 1, e.g. 0.2. If `threshold` is an absolute value, `get_mir()` returns only the miRNA names mentioned in at least `threshold` abstracts. If `threshold` is a float between 0 and 1, `get_mir()` returns only miRNA names mentioned in at least `threshold` abstracts of all abstracts. `threshold` requires the data frame to have a column with PubMed IDs.

If neither `top` nor `threshold` is set, `top` is automatically set to `5`.

Value

Character vector containing miRNA names.

See Also

Other get functions: `get_distinct_mir_df()`, `get_distinct_mir_vec()`, `get_pmid()`, `get_shared_mir_df()`, `get_shared_mir_vec()`, `get_snp()`

`get_pmid` *Get PubMed-IDs of a data frame*

Description

Get PubMed-IDs of a data frame.

Usage

```
get_pmid(df, col.pmid = PMID, copy = TRUE)
```

Arguments

`df`	Data frame containing PubMed-IDs.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`copy`	Boolean. If `copy = FALSE`, `get_pmid()` returns a character vector, containing PubMed-IDs. If `copy = TRUE`, `get_pmid()` copies PubMed-IDs to clipboard.

Details

Get PubMed-IDs of a data frame. `get_pmid` returns either a character vector, containing PubMed-IDs, or copies PubMed-IDs to clipboard. If PubMed-IDs are copied to the clipboard, they can be used e.g. to search for abstracts on PubMed.

Value

Copy to clipboard or character vector. If `copy = TRUE`, `get_pmid()` copies PubMed-IDs to clipboard. If `copy = FALSE`, `get_pmid()` returns a character vector, containing PubMed-IDs.

See Also

Other get functions: `get_distinct_mir_df()`, `get_distinct_mir_vec()`, `get_mir()`, `get_shared_mir_df()`, `get_shared_mir_vec()`, `get_snp()`

`get_shared_mir_df` *Get top miRNA names in common between two topics of a data frame*

Description

Get top miRNA names in common between two topics of a data frame.

Usage

```
get_shared_mir_df(
  df,
  top = 5,
  topic = NULL,
  col.topic = Topic,
  col.mir = miRNA,
  col.pmid = PMID
)
```

Arguments

`df`	Data frame containing at least two topics and miRNA names.
`top`	Integer. Number of top miRNA names to extract for both topics.
`topic`	String. Vector of strings containing topic names to compare miRNA names for. If `topic = NULL`, `topic` defaults to all topic names contained in `col.topic` in `df`. `topic` must only contain two topic names.
`col.topic`	Symbol. Column containing topic names.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.

Details

Get top miRNA names in common between two topics of a data frame. `get_shared_mir_df()` compares the top miRNA names of two topics in a data frame and returns the miRNA names in common.

Value

Character vector containing miRNA names in common between two topics.

See Also

Other get functions: `get_distinct_mir_df()`, `get_distinct_mir_vec()`, `get_mir()`, `get_pmid()`, `get_shared_mir_vec()`, `get_snp()`

`get_shared_mir_vec` *Get miRNA names in common between two vectors*

Description

Get miRNA names in common between two vectors.

Usage

```
get_shared_mir_vec(mirna.vec.1, mirna.vec.2)
```

Arguments

`mirna.vec.1`	Character vector. First vector containing miRNA names.
`mirna.vec.2`	Character vector. Second vector containing miRNA names.

Details

Get miRNA names in common between two vectors. `get_shared_mir_vec()` compares two vectors containing miRNA names and returns the miRNA names that are in both vectors.

Value

Character vector containing miRNA names in common between two vectors.

See Also

Other get functions: `get_distinct_mir_df()`, `get_distinct_mir_vec()`, `get_mir()`, `get_pmid()`, `get_shared_mir_df()`, `get_snp()`

`get_snp` *Get SNPs from a data frame*

Description

Get SNPs from a data frame.

Usage

```
get_snp(df, row = NULL, top = NULL, col.snp = SNPs, col.pmid = PMID)
```

Arguments

`df`	Data frame containing SNPs. If `top` is set, `df` must also contain PubMed IDs.
`row`	Integer. Optional. Specifies row from which SNP shall be obtained. Works best with a data frame listing counts only as from `count_snp()`. If neither `row` nor `top` is given, `row` is automatically set to `1`.
`top`	Integer. Optional. Specifies number of most frequent SNPs to return.
`col.snp`	Symbol. Column containing SNPs.
`col.pmid`	Symbol. Column containing PubMed IDs. Necessary if the data frame provided is not a count data frame.

Details

Get SNPs from a data frame.

- If a data frame containing SNP counts as from `count_snp()` is provided, these SNPs are specified by the row they are listed in. To get the SNPs by row, set the `row` argument.
- If a data frame with PubMed IDs is provided, these SNPs are specified by their top occurrence. To get the SNPs by frequency, set the `top` argument.

If neither `row` nor `top` is provided, `row` is automatically set to `1`.

Value

String or character vector containing SNPs.

See Also

`extract_snp()`, `count_snp()`, `subset_snp()`

Other get functions: `get_distinct_mir_df()`, `get_distinct_mir_vec()`, `get_mir()`, `get_pmid()`, `get_shared_mir_df()`, `get_shared_mir_vec()`

`indicate_mir` *Indicate if a miRNA name is contained in an abstract*

Description

Indicate if a miRNA name is contained in an abstract with "Yes"/"No".

Usage

```
indicate_mir(df, indicate.mir, col.mir = miRNA)
```

Arguments

`df`	Data frame containing miRNA names.
`indicate.mir`	Character vector. Vector containing miRNA names to indicate.
`col.mir`	Symbol. Column containing miRNA names.

Details

Indicate if a miRNA name is contained in an abstract with "Yes"/"No". This requires miRNA names already to be extracted, e.g. with `extract_mir_df()`, and to be stored in a separate column, specified by `col.mir`. `indicate_mir()` adds another column to a data frame which bears the name of the miRNA(s) of interest. Within this column, a "Yes" or "No" specifies if this miRNA name is contained in the corresponding abstract.

Value

Data frame with as many columns added as miRNA names given in `indicate.mir`. Per column, a "Yes" or "No" indicates if the miRNA name of interest is present in the corresponding abstract.

See Also

`extract_mir_df()`, `indicate_term()`

Other indicate functions: `indicate_term()`

`indicate_term` *Indicate if a term is contained in abstracts*

Description

Indicate if a term is contained in abstracts.

Usage

```
indicate_term(
  df,
  term,
  threshold = 1,
  case = FALSE,
  discard = FALSE,
  col.abstract = Abstract
)
```

Arguments

`df`	Data frame containing abstracts.
`term`	Character vector. Vector containing terms to indicate.
`threshold`	Integer. Sets how often a term must be in an abstract to be considered "present".
`case`	Boolean. If `case = TRUE`, strings contained in `term` are case sensitive. If `case = FALSE`, strings contained in `term` are case insensitive.
`discard`	Boolean. If `discard = TRUE`, only abstracts containing the terms in `term` are kept.
`col.abstract`	Symbol. Column containing abstracts.

Details

Indicate if a term is contained in an abstract. Terms provided can either be case sensitive or insensitive. Per term, a new column is added to the data frame indicating if the term is present in an abstract. Furthermore, if a term is considered "present" in an abstract can be regulated via the `threshold` argument. `threshold` determines how often a term must be in an abstract to be considered "present".

Value

Data frame. If `discard = FALSE`, the original data frame with additional columns per term is returned. If `discard = TRUE`, only abstracts containing the terms in `term` are returned.

See Also

`indicate_mir()`

Other indicate functions: `indicate_mir()`

`join_mirtarbase` *Add miRNA targets from miRTarBase version 8.0*

Description

Add miRNA targets from miRTarBase version 8.0 to a data frame.

Usage

```
join_mirtarbase(
  df,
  col.pmid.df = PMID,
  col.topic.df = NULL,
  filter_na = TRUE,
  reduce = FALSE
)
```

Arguments

`df`	Data frame containing PubMed-IDs that the miRNA targets shall be joined to.
`col.pmid.df`	Symbol. Column containing PubMed-IDs in `df`.
`col.topic.df`	Symbol. Optional. Only important if `reduce = TRUE`. If given, adds a topic column to the reduced data.frame.
`filter_na`	Boolean. If `filter_na = TRUE`, drops all rows containing `NA` in column `Target`.
`reduce`	Boolean. If `reduce = FALSE`, adds a new column containing miRNA targets to `df`. If `reduce = TRUE`, adds two new columns containing miRNA names and miRNA targets to `df`. All other columns except for the PubMed-ID column and (optionally) the topic column are dropped.

Details

Add miRNA targets from miRTarBase version 8.0 to a data frame. `join_mirtarbase()` can return two different data frames, regulated by `reduce`:

1. If `reduce = FALSE`, `join_mirtarbase()` adds targets from miRTarBase 8.0 to the data frame in a new column. These targets then correspond to the targets determined in the research paper, but do not necessarily correspond to the miRNA names mentioned in the abstract.
2. If `reduce = TRUE`, `join_mirtarbase()` adds targets from miRTarBase 8.0 to the data frame in a new column. However, an altered data frame is returned, containing the PubMed-IDs, targets, and miRNAs from miRTarBase 8.0.

miRTarBase was published in

Hsi-Yuan Huang, Yang-Chi-Dung Lin, Jing Li, et al., miRTarBase 2020: updates to the experimentally validated microRNA–target interaction database, Nucleic Acids Research, Volume 48, Issue D1, 08 January 2020, Pages D148–D154, https://doi.org/10.1093/nar/gkz896

Value

Data frame containing miRNA targets.

See Also

Other target functions: `count_target()`, `join_targets()`, `plot_target_count()`, `plot_target_mir_scatter()`

`join_targets` *Add miRNA targets from an xlsx-file to a data frame*

Description

Add miRNA targets from an external xlsx-file to a data frame.

Usage

```
join_targets(
  df,
  excel_file,
  col.pmid.excel,
  col.target.excel,
  col.mir.excel = NULL,
  col.pmid.df = PMID,
  col.topic.df = NULL,
  filter_na = TRUE,
  stem_mir_excel = TRUE,
  reduce = FALSE
)
```

Arguments

`df`	Data frame containing PubMed-IDs that the miRNA targets shall be joined to.
`excel_file`	xlsx-file. xlsx-file containing miRNA targets and PubMed-IDs.
`col.pmid.excel`	String. Column containing PubMed-IDs of the `excel_file`.
`col.target.excel`	String. Column containing targets of the `excel_file`.
`col.mir.excel`	String. Optional. Column containing miRNAs of the `excel_file`. Needed if `reduce = TRUE`.
`col.pmid.df`	Symbol. Column containing PubMed-IDs in `df`.
`col.topic.df`	Symbol. Optional. Only important if `reduce = TRUE`. If given, adds a topic column to the reduced data.frame.
`filter_na`	Boolean. If `filter_na = TRUE`, drops all rows containing `NA` in column `Target`.
`stem_mir_excel`	Boolean. If `stem_mir_excel = TRUE`, miRNA names provided in `col.mir.excel` are reduced to their stem, e.g. "miR-20a-5p" becomes "miR-20".

`reduce`	Boolean. If `reduce = FALSE`, adds a new column containing miRNA targets to `df`. If `reduce = TRUE`, adds two new columns containing miRNA names and miRNA targets to `df`. All other columns except for the PubMed-ID column and (optionally) the topic column are dropped.

Details

Add miRNA targets from an external xlsx-file to a data frame. To add the targets to the data frame, the xlsx-file and the data frame need to have one column in common, such as PubMed-IDs. `join_targets()` can return two different data frames, regulated by `reduce`:

1. If `reduce = FALSE`, `join_targets()` adds targets from an excel-file to the data frame in a new column. These targets then correspond to the targets determined in the research paper, but do not necessarily correspond to the miRNA names mentioned in the abstract.
2. If `reduce = TRUE`, `join_targets()` adds targets from an xlsx-file to the data frame in a new column. However, an altered data frame is returned, containing the PubMed-IDs, targets, and miRNAs from the excel-file. For `reduce = TRUE` to work, the xlsx-file provided must contain a column with miRNA names.

Value

Data frame containing miRNA targets.

See Also

Other target functions: `count_target()`, `join_mirtarbase()`, `plot_target_count()`, `plot_target_mir_scatter()`

`ngram_stopwords` *Stop words for n-grams*

Description

Vector containing stop words for n-grams, based on tidytext::stop_words.

Usage

```
ngram_stopwords
```

Format

Character vector.

Source

tidytext::stop_words

`patients_keywords` *Keywords - patients.*

Description

Keywords to identify abstracts investigating miRNAs in patients.

Usage

```
patients_keywords
```

Format

An object of class `character` of length 10.

`plot_lda_term` *Plot terms associated with LDA-fitted topics*

Description

Plot terms associated with LDA-fitted topics.

Usage

```
plot_lda_term(lda_model, top.terms = 10, title = NULL)
```

Arguments

`lda_model`	LDA-model.
`top.terms`	Integer. Top terms to plot per topic.
`title`	String. Plot title.

Details

Plot terms associated with LDA-fitted topics. For each topic in the LDA-model, the top terms are plotted. Plotting `top.terms` for each topic can help identifying its subject.

Value

Bar plot with top terms per topic.

See Also

Other LDA functions: `assign_topic_lda()`, `fit_lda()`, `plot_perplexity()`

`plot_mir_count` *Plot count of most frequently mentioned miRNA names*

Description

Plot count of most frequently mentioned miRNA names in a data frame.

Usage

```
plot_mir_count(
  df,
  top = 10,
  colour = "steelblue3",
  col.mir = miRNA,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names.
`top`	Integer. Specifies number of most frequent miRNA names to plot.
`colour`	String. Colour of bar plot.
`col.mir`	Symbol. Column containing miRNA names.
`title`	String. Plot title.

Details

Plot count of most frequently mentioned miRNA names in a data frame. How many most frequently mentioned miRNAs are plotted is determined via the `top` argument. Ties among the most frequently mentioned miRNAs are treated as the same rank, e.g. if *miR-126*, *miR-34*, and *miR-29* were all mentioned the most often, they would all be plotted by specifying `top = 1`, `top = 2`, or `top = 3`.

Value

Bar plot with the most frequently mentioned miRNAs names in `df`.

See Also

`count_mir()`, `count_mir_threshold()`, `plot_mir_count_threshold()`

Other count functions: `count_mir_threshold()`, `count_mir()`, `count_snp()`, `plot_mir_count_threshold()`

`plot_mir_count_threshold` *Plot occurrence count of miRNA names over different thresholds*

Description

Plot occurrence count of distinct miRNA names over different thresholds.

Usage

```
plot_mir_count_threshold(
  df,
  start = 1,
  end = 5,
  bins = NULL,
  colour = "steelblue3",
  col.mir = miRNA,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing columns with miRNAs and PubMed-IDs.
`start`	Integer or float. Must be greater than 0 and smaller than `end`.
`end`	Integer or float. Must be greater than 0 and greater than `start`. If `start` >= 1, `plot_mir_count_threshold()` plots number of miRNAs above different absolute thresholds, ranging from `start` to `end`. If `start` >= 0 and `end` <= 1, `bins` must be specified. If `bins` is not specified, `bins` is automatically set to `10`. `plot_mir_count_threshold()` then plots number of miRNAs above different thresholds, ranging from `start` to `end` in n `bins`. If `start` >= 0 and `end` <= 1 and the value of `start` is too low for the number of miRNAs to be plotted, `plot_mir_count_threshold()` raises a warning, suggesting a more appropriate `start` value.
`bins`	Integer. Optional. Only necessary if `start` >=0 and `end` <=1. Specifies number of bins between `start` and `end`. If `start` >= 0, `end` <= 1, and `bins` is not specified, `bins` is automatically set to `10`.
`colour`	String. Colour of bar plot.
`col.mir`	Symbol. Column containing miRNAs.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Plot occurrence of distinct miRNA names over different thresholds. These thresholds can either be absolute values or floating values between 0 and 1. If the thresholds are absolute values, number of distinct miRNA names mentioned in at least n abstracts are plotted, where n is the range of thresholds defined by `start` and `end`. If the thresholds are floating values, `bins` must be specified as well. Then the umber of distinct miRNA names mentioned in at least n abstracts over `bins` are plotted, where n is the range of thresholds between `start` and `end`. Overall, plotting can help in identifying if the abstracts at hand mention different miRNAs in a balanced way, or if there are few miRNAs dominating the field.

Value

Bar plot counting the occurrence of miRNA names above different thresholds.

See Also

`count_mir_threshold()`, `count_mir()`, `plot_mir_count()`

Other count functions: `count_mir_threshold()`, `count_mir()`, `count_snp()`, `plot_mir_count()`

`plot_mir_development` *Plot development of miRNA name mentioning over time*

Description

Plot development of miRNA name mentioning over time.

Usage

```
plot_mir_development(
  df,
  mir,
  start = NULL,
  end = NULL,
  linetype = "miRNA",
  alpha = 0.8,
  width = 0.3,
  col.mir = miRNA,
  col.year = Year,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names and publication years.
`mir`	Character vector. Vector containing miRNA names to plot.

`start`	Numeric. Optional. Specifies start year. If `start = NULL`, `start` is set to the oldest year in `df`.
`end`	Numeric. Optional. Specifies end year. If `end = NULL`, `end` is set to the youngest year in `df`.
`linetype`	String. Specifies linetype. `linetype` can take on values as mentioned in the geom_line documentation of **ggplot2**. Additionally, `linetype` can be set to `"miRNA"`. If `linetype = "miRNA"`, each miRNA name in `mir` has its own linetype.
`alpha`	Float. Opacity of lines.
`width`	Float. Width of dodging lines.
`col.mir`	Symbol. Column containing miRNA names.
`col.year`	Symbol. Column containing year.
`title`	String. Plot title.

Details

Plot how often a miRNA name was mentioned per year.

Value

Line plot displaying how often a miRNA name was mentioned per year..

See Also

Other miR development functions: `plot_mir_new()`

`plot_mir_new` *Plot number of newly mentioned miRNA names/year*

Description

Plot number of newly mentioned miRNA names/year.

Usage

```
plot_mir_new(
  df,
  threshold = 1,
  start = NULL,
  end = NULL,
  colour = "steelblue3",
  col.mir = miRNA,
  col.year = Year,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names and publication years.
`threshold`	Integer. Specifies how often a miRNA must be mentioned in a year to be considered "mentioned".
`start`	Integer. Optional. Beginning of publication period. If `start = NULL`, `start` is set to the least recent year in `df`.
`end`	Integer. Optional. End of publication period. If `end = NULL`, `end` is set to the most recent year in `df`.
`colour`	String. Colour of bar plot.
`col.mir`	Symbol. Column containing miRNA names.
`col.year`	Symbol. Column containing publication year.
`title`	String. Plot title.

Details

Plot how many miRNAs are mentioned for the first time in different year. If a miRNA is considered to be "mentioned" in a year can be regulated via the `threshold` argument. If, for example, `threshold` is set to 3, but a miRNA is mentioned only twice in a year, it is not considered to be "mentioned" for this year.

Value

Bar plot displaying the number of newly mentioned miRNA names/year.

See Also

Other miR development functions: `plot_mir_development()`

`plot_mir_terms` *Plot count of top terms associated with a miRNA name*

Description

Plot count of top terms associated with a miRNA name.

Usage

```
plot_mir_terms(
  df,
  mir,
  top = 20,
  tf.idf = FALSE,
  token = "words",
  ...,
  stopwords = stopwords_miretrieve,
```

```
  stopwords_ngram = TRUE,
  normalize = TRUE,
  colour = "steelblue3",
  col.mir = miRNA,
  col.abstract = Abstract,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing miRNA names, abstracts, and PubMed-IDs.
`mir`	String. miRNA name of interest.
`top`	Integer. Number of top terms to plot.
`tf.idf`	Boolean. If `tf.idf = TRUE`, terms are weighed in a tf-idf fashion. miRNA names are considered as separate documents and terms often associated with one miRNA, but not with other miRNAs get more weight.
`token`	String. Specifies how abstracts shall be split up. Taken from `unnest_tokens()` in the **tidytext** package: "Unit for tokenizing, or a custom tokenizing function. Built-in options are "words" (default), "characters", "character_shingles", "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", "regex", (...), and "ptb" (Penn Treebank). If a function, should take a character vector and return a list of character vectors of the same length."
`...`	Additional arguments for tokenization, if necessary.
`stopwords`	Data frame containing stop words.
`stopwords_ngram`	Boolean. Specifies if stop words shall be removed from abstracts when using ngrams. Only applied when `token = 'ngrams'`.
`normalize`	Boolean. If `normalize = TRUE`, normalizes the number of abstracts to the total number of abstracts with a miRNA name in a topic. Cannot be applied with `tf.idf = TRUE`.
`colour`	String. Colour of bar plot.
`col.mir`	Symbol. Column containing miRNA names
`col.abstract`	Symbol. Column containing abstracts.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Title plot.

Details

Plot count of top terms associated with a miRNA name. Top terms associated with `mir` have to be in `df` as abstracts. Number of top terms to plot is regulated via the `top` argument. Terms can either be evaluated as their count or in a tf-idf fashion. If terms are evaluated as their count, they can either be evaluated as their raw count, e.g. in how many abstracts they are mentioned in conjunction with the miRNA name, or as their relative count, e.g. in how many abstracts containing the miRNA they are mentioned compared to all abstracts containing the miRNA. If terms are evaluated in a tf-idf fashion, miRNA names are considered as separate documents and terms often associated with one miRNA, but not with other miRNAs get more weight. `plot_mir_terms()` is based on the tools available in the **tidytext** package.

Value

Bar plot displaying the count of the top terms associated with a miRNA name.

See Also

`plot_wordcloud()`, `tidytext::unnest_tokens()`

Other miR term functions: `plot_wordcloud()`

`plot_perplexity` *Plot perplexity score of various LDA models*

Description

Plot perplexity score of various LDA models.

Usage

```
plot_perplexity(
  df,
  start = 2,
  end = 5,
  stopwords = stopwords_miretrieve,
  method = "gibbs",
  control = NULL,
  col.abstract = Abstract,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing abstracts and PubMed-IDs.
`start`	Integer. Minimum amount of `k` topics for the LDA model to fit. Must be >=2.
`end`	Integer. Maximum amount of `k` topics for the LDA model to fit.
`stopwords`	Data frame containing stop words.
`method`	String. Either `"gibbs"` or `"VEM"`.
`control`	Control parameters for LDA modeling. For more information, see the documentation of the `LDAcontrol` class in the **topicmodels** package.
`col.abstract`	Column containing abstracts.
`col.pmid`	Column containing PubMed-ID.
`title`	String. Plot title.

Details

Plot perplexity score of various LDA models. `plot_perplexity()` fits different LDA models for `k` topics in the range between `start` and `end`. For each LDA model, the perplexity score is plotted against the corresponding value of `k`. Plotting the perplexity score of various LDA models can help in identifying the optimal number of topics to fit an LDA model for. `plot_perplexity()` is based on `LDA()` from the package **topicmodels**.

Value

Elbow plot displaying perplexity scores of different LDA models.

See Also

`fit_lda()`

Other LDA functions: `assign_topic_lda()`, `fit_lda()`, `plot_lda_term()`

`plot_score_animals` *Plot frequency of animal model scores in abstracts*

Description

Plot frequency of animal model scores in abstracts.

Usage

```
plot_score_animals(
  df,
  keywords = animal_keywords,
  case = FALSE,
  bins = NULL,
  colour = "steelblue3",
  col.abstract = Abstract,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. The animal model score is calculated based on these keywords. How much weight a keyword in `keywords` carries is determined how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score.
`case`	Boolean. If `case = TRUE`, terms contained in `keywords` are case sensitive. If `case = FALSE`, terms contained in `keywords` are case insensitive.

`bins`	Integer. Specifies how many bins are used to plot the distribution. If `bins = NULL`, bins are calculated over the whole range of scores, with one bin per score.
`colour`	String. Colour of histogram.
`col.abstract`	Symbol. Column containing abstracts.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Plots a frequency distribution of animal model scores in abstracts of a data frame. The animal model score is influenced by the choice of terms in `keywords`. Plotting the distribution can help deciding if the terms are well-chosen, or in choosing the right threshold to decide which abstracts are considered to contain animal models.

Value

Histogram displaying the distribution of animal scores in abstracts.

See Also

`calculate_score_animals()`

Other score functions: `assign_topic()`, `calculate_score_animals()`, `calculate_score_biomarker()`, `calculate_score_patients()`, `calculate_score_topic()`, `plot_score_biomarker()`, `plot_score_patients()`, `plot_score_topic()`

`plot_score_biomarker` *Plot frequency of biomarker scores in abstracts*

Description

Plot frequency of biomarker scores in abstracts.

Usage

```
plot_score_biomarker(
  df,
  keywords = biomarker_keywords,
  case = FALSE,
  bins = NULL,
  colour = "steelblue3",
  col.abstract = Abstract,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. The biomarker score is calculated based on these keywords. How much weight a keyword in `keywords` carries is determined how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score.
`case`	Boolean. If `case = TRUE`, terms contained in `keywords` are case sensitive. If `case = FALSE`, terms contained in `keywords` are case insensitive.
`bins`	Integer. Specifies how many bins are used to plot the distribution. If `bins = NULL`, bins are calculated over the whole range of scores, with one bin per score.
`colour`	String. Colour of histogram.
`col.abstract`	Symbol. Column containing abstracts.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Plots a frequency distribution of biomarker scores in abstracts of a data frame. The biomarker score is influenced by the choice of terms in `keywords`. Plotting the distribution can help deciding if the terms are well-chosen, or in choosing the right threshold to decide which abstracts are considered to contain use of miRNAs as biomarker.

Value

Histogram displaying the distribution of biomarker scores in abstracts.

See Also

`calculate_score_biomarker()`

Other score functions: `assign_topic()`, `calculate_score_animals()`, `calculate_score_biomarker()`, `calculate_score_patients()`, `calculate_score_topic()`, `plot_score_animals()`, `plot_score_patients()`, `plot_score_topic()`

`plot_score_patients` *Plot frequency of patient scores in abstracts*

Description

Plot frequency of patient scores in abstracts.

Usage

```
plot_score_patients(
  df,
  keywords = patients_keywords,
  case = FALSE,
  bins = NULL,
  colour = "steelblue3",
  col.abstract = Abstract,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. The score is calculated based on these keywords. How much weight a keyword in `keywords` carries is determined how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score.
`case`	Boolean. If `case = TRUE`, terms contained in `keywords` are case sensitive. If `case = FALSE`, terms contained in `keywords` are case insensitive.
`bins`	Integer. Specifies how many bins are used to plot the distribution. If `bins = NULL`, bins are calculated over the whole range of scores, with one bin per score.
`colour`	String. Colour of histogram.
`col.abstract`	Symbol. Column containing abstracts.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Plots a frequency distribution of patient scores in abstracts of a data frame. The patient score is influenced by the choice of terms in `keywords`. Plotting the distribution can help deciding if the terms are well-chosen, or in choosing the right threshold to decide which abstracts are considered to contain patient material

Value

Histogram displaying the distribution of patient scores in abstracts.

See Also

`calculate_score_patients()`

Other score functions: `assign_topic()`, `calculate_score_animals()`, `calculate_score_biomarker()`, `calculate_score_patients()`, `calculate_score_topic()`, `plot_score_animals()`, `plot_score_biomarker()`, `plot_score_topic()`

`plot_score_topic` *Plot frequency of self-chosen topic scores in abstracts*

Description

Plot frequency of self-chosen topic scores in abstracts.

Usage

```
plot_score_topic(
  df,
  keywords,
  case = FALSE,
  name.topic = "TOPIC",
  bins = NULL,
  colour = "steelblue3",
  col.abstract = Abstract,
  col.pmid = PMID,
  title = NULL
)
```

Arguments

`df`	Data frame containing abstracts.
`keywords`	Character vector. Vector containing keywords. How much weight a keyword in `keywords` carries is determined by how often it is present in `keywords`, e.g. if a keyword is mentioned twice in `keywords` and it is mentioned only once in an abstract, it adds 2 points to the score.
`case`	Boolean. If `case = TRUE`, terms contained in `keywords` are case sensitive. If `case = FALSE`, terms contained in `keywords` are case insensitive.
`name.topic`	String. Name of the topic.
`bins`	Integer. Specifies how many bins are used to plot the distribution. If `bins = NULL`, bins are calculated over the whole range of scores, with one bin per score.
`colour`	String. Colour of histogram.
`col.abstract`	Symbol. Column containing abstracts.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.

Details

Plots a frequency distribution of self-chosen topic scores in abstracts of a data frame. The topic score is influenced by the choice of terms in `keywords`. Plotting the distribution can help in choosing the right threshold to decide which abstracts correspond to the self-chosen topic.

Value

Histogram displaying the distribution of self-chosen topic scores in abstracts.

See Also

`calculate_score_topic()`, `assign_topic()`

Other score functions: `assign_topic()`, `calculate_score_animals()`, `calculate_score_biomarker()`, `calculate_score_patients()`, `calculate_score_topic()`, `plot_score_animals()`, `plot_score_biomarker()`, `plot_score_patients()`

`plot_target_count` *Plot count of miRNA targets*

Description

Plot count of miRNA targets.

Usage

```
plot_target_count(
  df,
  top = NULL,
  threshold = NULL,
  colour = "steelblue3",
  col.target = Target,
  title = NULL
)
```

Arguments

`df`	Data frame with miRNA targets.
`top`	Numeric. Specifies number of top targets to be plotted.
`threshold`	Numeric. Specifies how often a target must be in `col.target` to be plotted.
`colour`	String. Colour of bar plot.
`col.target`	Symbol. Column containing miRNA targets.
`title`	String. Plot title.

Details

Plot count of miRNA targets as a bar plot. How many targets are plotted is determined either by the `top` or by the `threshold` argument. If `top` is given, targets with the highest count are plotted. Ties among targets with the highest count are treated as the same rank, e.g. if *PTEN*, *AKT*, and *VEGFA* all had the highest count, they would all be plotted by specifying `top = 1`, `top = 2`, and `top = 3`. If `threshold` is given, only targets with a count of at least `threshold` are plotted. If neither `top` nor `threshold` is given, `top` is automatically set to `5`.

Value

Bar plot with target counts.

See Also

`count_target()`, `join_targets()`

Other target functions: `count_target()`, `join_mirtarbase()`, `join_targets()`, `plot_target_mir_scatter()`

`plot_target_mir_scatter`

Plot targets and corresponding miRNAs as a scatter plot

Description

Plot targets and corresponding miRNAs as a scatter plot.

Usage

```
plot_target_mir_scatter(
  df,
  mir = NULL,
  target = NULL,
  top = NULL,
  threshold = NULL,
  filter_for = "target",
  col.target = Target,
  col.mir = miRNA,
  col.topic = Topic,
  col.pmid = PMID,
  title = NULL,
  height = 0.05,
  width = 0.05,
  alpha = 0.6
)
```

Arguments

`df`	Data frame containing targets and miRNA names.
`mir`	String or character vector. Specifies which miRNAs to plot.
`target`	String or character vector. Specifies which targets to plot.
`top`	Numeric. Specifies number of top targets/miRNA names to be plotted.
`threshold`	Numeric. Specifies how often a target/miRNA name must be in `df` to be plotted.
`filter_for`	String. Must either be `"target"` or `"miRNA"`. Specifies if `threshold`/`top` shall be applied to targets or miRNA names.
`col.target`	Symbol. Column containing miRNA targets.

`col.mir`	Symbol. Column containing miRNA names.
`col.topic`	Symbol. Column containing topic names.
`col.pmid`	Symbol. Column containing PubMed-IDs.
`title`	String. Plot title.
`height`	Double. Specifies height of jitter.
`width`	Double. Specifies width of jitter.
`alpha`	Double. Specifies opacity of points.

Details

Plot targets and corresponding miRNAs as a scatter plot. With `filter_for`, it can be determined if the focus shall be on the top targets to plot their corresponding miRNAs, or if the focus shall be on the top miRNA names to plot their corresponding targets. What "top targets" or "top miRNA names" mean can be determined via the `top` and `threshold` arguments.

- If `top` is given, `df` is filtered for the most frequent targets/miRNA names.
- If `threshold` is given, data frame is filtered for all targets/miRNA names mentioned at least `threshold` times.
- If neither `top` nor `threshold` is given, `top` is automatically set to `5`.

By plotting miRNAs against their targets, it is visualized if one miRNA regulates many targets, or if one target is regulated by many miRNAs. Furthermore, the miRNA-target interactions are labelled according to their topic in `col.topic`, thereby facilitating comparison of miRNA-target interactions across different topics.

Value

Scatter plot with targets and corresponding miRNAs.

See Also

`join_targets()`

Other target functions: `count_target()`, `join_mirtarbase()`, `join_targets()`, `plot_target_count()`

`plot_wordcloud` *Create wordcloud of terms associated with a miRNA name*

Description

Create wordcloud of terms associated with a miRNA name.

Usage

```
plot_wordcloud(
  df,
  mir,
  min.freq = 1,
  max.terms = 20,
  tf.idf = FALSE,
  token = "words",
  ...,
  stopwords = stopwords_miretrieve,
  stopwords_ngram = TRUE,
  colours = "black",
  random.colour = TRUE,
  ordered.colour = FALSE,
  col.mir = miRNA,
  col.abstract = Abstract,
  col.pmid = PMID
)
```

Arguments

`df`	Data frame containing miRNA names, abstracts, and PubMed-IDs.
`mir`	String. miRNA name of interest.
`min.freq`	Integer. Specifies least number of times a term must be associated with `mir` to be plotted.
`max.terms`	Integer. Maximum number of terms to plot.
`tf.idf`	Boolean. If `tf.idf = TRUE`, terms are weighed in a tf-idf fashion. miRNA names are considered as separate documents, and terms often associated with one miRNA, but not with other miRNAs get more weight. Cannot be used if `normalize = TRUE`. If `tf.idf = TRUE` and `normalize = TRUE`, `tf.idf = TRUE` is ignored.
`token`	String. Specifies how abstracts shall be split up. Taken from `unnest_tokens()` in the **tidytext** package: "Unit for tokenizing, or a custom tokenizing function. Built-in options are "words" (default), "characters", "character_shingles", "ngrams", "skip_ngrams", "sentences", "lines", "paragraphs", "regex", (...), and "ptb" (Penn Treebank). If a function, should take a character vector and return a list of character vectors of the same length."
`...`	Additional arguments for tokenization, if necessary.
`stopwords`	Data frame containing stop words.
`stopwords_ngram`	Boolean. Specifies if stop words shall be removed from abstracts when using ngrams. Only applied when `token = 'ngrams'`.
`colours`	Vector of strings. Colours for wordcloud.
`random.colour`	Boolean. Taken from `wordcloud()` in the **wordcloud** package: "Choose colours randomly from `colours`. If false, the colour is chosen based on the frequency."

`ordered.colour`	Boolean. Taken from `wordcloud()` in the **wordcloud** package: "If true, then colours are assigned to words in order."
`col.mir`	Symbol. Column containing miRNA names.
`col.abstract`	Symbol. Column containing abstracts.
`col.pmid`	Symbol. Column containing PubMed-IDs.

Details

Create wordcloud of terms associated with a miRNA name. miRNA names must be in a data frame `df`, while terms are taken from abstracts contained in `df`. Number of terms to plot is regulated by `max.terms`, while `min.freq` regulates the least number of times a term must be mentioned to be plotted. Terms can either be evaluated as their raw count, e.g. how often they are mentioned in conjunction with the miRNA of interest, or weighed in a tf-idf fashion. If `tf.idf = TRUE`, miRNA names are considered as separate documents, and terms often associated with one miRNA, but not with other miRNAs get more weight. `plot_wordcloud()` is based on the tools available in the **wordcloud** package.

Value

Wordcloud of terms associated with a miRNA name.

See Also

`plot_mir_terms()`, `wordcloud::wordcloud()`, `tidytext::unnest_tokens()`

Other miR term functions: `plot_mir_terms()`

`read_pubmed` *Convert PubMed-file from PubMed into a data frame*

Description

Convert PubMed-file from PubMed into a data frame.

Usage

```
read_pubmed(pubmed_file, topic = NULL)
```

Arguments

`pubmed_file`	PubMed-file as .txt, downloaded from PubMed.
`topic`	String. Optional. If provided, adds a "Topic" column containing `topic`.

Details

Convert an PubMed-file from PubMed into a data frame. The PubMed-file should contain PubMed-IDs, abstracts from research articles, abstract title, publication year, abstract language, and article type. The data frame created holds at least six columns, namely

- `PMID`, containing the PubMed-ID,
- `Year`, containing the publication year,
- `Title`, containing the title of the abstracts,
- `Abstract`, containing the actual abstract,
- `Language`, containing the language(s) of the paper,
- `Type`, containing the article type.

If `topic` is provided, a "Topic" column is added, assigning all abstracts in `df` to `topic`.

`read_pubmed()` is faster than `read_pubmed_jats()` and thus recommended.

Value

Data frame containing PubMed-IDs, abstracts, abstract titles, publication years, languages, and article types.

See Also

`read_pubmed_jats()`

Other external data functions: `read_pubmed_jats()`, `save_excel()`, `save_plot()`

`read_pubmed_jats` *Convert JATS-file from PubMed into a data frame*

Description

Convert JATS-file from PubMed into a data frame.

Usage

```
read_pubmed_jats(jats_file, topic = NULL)
```

Arguments

`jats_file`	JATS-file, downloaded from PubMed.
`topic`	String. Optional. If provided, adds a "Topic" column containing `topic`.

Details

Converts an JATS-file from PubMed into a data frame. The JATS-file should contain PubMed-IDs, abstracts from research articles, abstract title, publication year, abstract language, and article type. The data frame created holds at least six columns, namely

- `PMID`, containing the PubMed-ID,
- `Year`, containing the publication year,
- `Title`, containing the title of the abstracts,
- `Abstract`, containing the actual abstract,
- `Language`, containing the language(s) of the paper,
- `Type`, containing the article type.

If `topic` is provided, a "Topic" column is added, assigning all abstracts in `df` to `topic`.

`read_pubmed()` is faster than `read_pubmed_jats()` and thus recommended.

Value

Data frame containing PubMed-IDs, abstracts, abstract titles, publication years, languages, and article types.

See Also

`read_pubmed()`

Other external data functions: `read_pubmed()`, `save_excel()`, `save_plot()`

`save_excel` *Save data frame(s) as xlsx-file*

Description

Save data frame(s) locally as an xlsx-file.

Usage

```
save_excel(..., excel_file = "miRetrieve_data.xlsx")
```

Arguments

`...`	Data frame(s) to save.
`excel_file`	String. File name that `...` shall be saved to. Must end in ".xlsx".

Details

Saves data frame locally as an xlsx-file. If more than one data frame is provided, data frames are saved in an xlsx-file with one sheet per data frame.

Wrapper function of `write.xlsx()` from **openxlsx**.

Value

xlsx-file, locally saved.

See Also

`openxlsx::write.xlsx()`

Other external data functions: `read_pubmed_jats()`, `read_pubmed()`, `save_plot()`

`save_plot` *Save the last generated figure*

Description

Save the last generated figure locally.

Usage

```
save_plot(
  plot_file,
  width = NULL,
  height = NULL,
  units = "in",
  dpi = 300,
  device = NULL
)
```

Arguments

`plot_file`	String. File name that the figure shall be saved to. Can end in either ".png", ".tiff", ".pdf", ".jpeg", or ".bmp". For more information, see the documentation of `ggplot2::ggsave()`.
`width`	Integer. Optional. Plot width. If `width = NULL`, `width` is set to the width of the plotting window.
`height`	Integer. Optional. Plot height If `height = NULL`, `height` is set to the height of the plotting window.
`units`	String. Units for `width` and `height`.
`dpi`	Integer. Resolution for raster graphics such as .pdf-files.
`device`	String or function. Specifies which device to use (such as "pdf" or `cairo_pdf`)

Details

Saves the last generated figure locally. Wrapper function of `ggsave()` from **ggplot2**. For further details, please see ?ggplot2::ggsave.

Value

Plot, locally saved.

See Also

`ggplot2::ggsave()`

Other external data functions: `read_pubmed_jats()`, `read_pubmed()`, `save_excel()`

`stopwords_2gram` *Stop words for text mining with common PubMed 2-grams*

Description

Data frame containing PubMed 2-gram stop words, manually curated from PubMed abstracts

Usage

```
stopwords_2gram
```

Format

Tibble.

- `word`: Column containing stop words. Pulled from various PubMed abstracts.
- `lexicon`: Column specifying lexicon.

Source

Manually created from various PubMed abstracts.

`stopwords_miretrieve` *Stop words for text mining with miRetrieve*

Description

Data frame containing English stop words, PubMed stop words, and common 2-gram stopwords. English stop words are based on tidytext::stop_words, while PubMed stop words are manually curated from PubMed abstracts

Usage

```
stopwords_miretrieve
```

Format

Tibble.

- `word`: Column containing stop words. Pulled from various PubMed abstracts.
- `lexicon`: Column specifying lexicon.

Source

tidytext::stop_words; manually created from various PubMed abstracts.

`stopwords_pubmed` *Stop words for text mining from PubMed abstracts*

Description

Data frame containing PubMed stop words, manually curated from PubMed abstracts

Usage

```
stopwords_pubmed
```

Format

Tibble.

- `word`: Column containing stop words. Pulled from various PubMed abstracts.
- `lexicon`: Column specifying lexicon.

Source

Manually created from various PubMed abstracts.

`subset_df` *Subset data frame for a term*

Description

Subset data frame for a term in a specified column.

Usage

```
subset_df(df, col.filter, filter_for = "Yes")
```

Arguments

`df`	Data frame to subset.
`col.filter`	String. Name of column to filter.
`filter_for`	String. Term to filter for.

Details

Subset data frame for a term in a specified column. `subset_df()` filters a data frame for a certain term in a specified column. All rows containing the term in the specified column are kept, while the other rows are silently dropped. Here, `col.filter` is a string rather than a symbol to facilitate filtering in columns that carry special characters such as '-' in their name.

Value

Data frame, subset for rows where `filter_for` was present in `col.filter`.

See Also

`indicate_term()`, `indicate_mir()`, `extract_snp()`

Other subset functions: `subset_mir_threshold()`, `subset_mir()`, `subset_research()`, `subset_review()`, `subset_snp()`, `subset_year()`

`subset_mir` *Subset data frame for specific miRNA names*

Description

Subset data frame for specific miRNA names only.

Usage

```
subset_mir(df, mir.retain, col.mir = miRNA)
```

Arguments

`df`	Data frame containing a miRNA names.
`mir.retain`	Character vector. Vector specifying which miRNA names to keep. miRNA names in `mir.retain` must match miRNA names in `col.mir` in `df`.
`col.mir`	Symbol. Column containing miRNA names.

Details

Subset data frame for specific miRNA names only.

Value

Data frame containing only specified miRNA names. If no miRNA name in `mir.retain` matches a miRNA name in `col.mir`, `subset_mir()` stops with a warning saying *"No miRNA name in 'mir.retain' matches a miRNA name in 'col.mir'. Could not filter for miRNA name."*.

See Also

`get_mir()`, `subset_mir_threshold()`

Other subset functions: `subset_df()`, `subset_mir_threshold()`, `subset_research()`, `subset_review()`, `subset_snp()`, `subset_year()`

`subset_mir_threshold` *Subset data frame for miRNA names exceeding a threshold*

Description

Subset data frame for miRNA names whose frequency exceeds a threshold.

Usage

```
subset_mir_threshold(df, threshold = 1, col.mir = miRNA, col.pmid = PMID)
```

Arguments

`df`	Data frame containing miRNA names and a PubMed-IDs.
`threshold`	Integer or float. If `threshold` >= 1, retains miRNA names in at least `threshold` abstracts. If `threshold` is between 0 and 1, retains miRNA names mentioned in at least `threshold` abstracts of all abstracts in `df`.
`col.mir`	Symbol. Column containing miRNA names.
`col.pmid`	Symbol. Column containing PubMed-IDs.

Details

Subset data frame for miRNA names whose frequency exceeds a threshold. This threshold can either be an absolute value, e.g. 3, or a float between 0 and 1, e.g. 0.2. If `threshold` is an absolute value, `subset_mir_threshold()` retains miRNA names mentioned in at least `threshold` abstracts. If `threshold` is a float between 0 and 1, `subset_mir_threshold()` retains miRNA names mentioned in at least `threshold` abstracts of all abstracts in `df`.

Value

Data frame, subset for miRNA names whose frequency exceeds a threshold.

See Also

`get_mir()`, `subset_mir()`

Other subset functions: `subset_df()`, `subset_mir()`, `subset_research()`, `subset_review()`, `subset_snp()`, `subset_year()`

`subset_research` *Subset data frame for abstracts of research articles*

Description

Subset data frame for abstracts of research articles only.

Usage

```
subset_research(df, col.type = Type)
```

Arguments

`df`	Data frame containing article types.
`col.type`	Symbol. Column containing articles types.

Details

Subset data frame for abstracts of research articles only. At the same time, abstracts from other article types such as *Review*, *Letter*, etc. are dropped.

Value

Data frame containing abstracts of research articles only.

See Also

`subset_review()`, `subset_year()`

Other subset functions: `subset_df()`, `subset_mir_threshold()`, `subset_mir()`, `subset_review()`, `subset_snp()`, `subset_year()`

`subset_review` *Subset data frame for abstracts of review articles*

Description

Subset data frame for abstracts of review articles only.

Usage

```
subset_review(df, col.type = Type)
```

Arguments

`df`	Data frame containing article types.
`col.type`	Symbol. Column containing articles types.

Details

Subset data frame for abstracts of review articles only. At the same time, abstracts from other article types such as *Journal Article*, *Letter*, etc. are dropped.

Value

Data frame containing abstracts of review articles only.

See Also

`subset_research()`, `subset_year()`

Other subset functions: `subset_df()`, `subset_mir_threshold()`, `subset_mir()`, `subset_research()`, `subset_snp()`, `subset_year()`

`subset_snp` *Subset data frame for specific SNPs*

Description

Subset data frame for specific SNPs only.

Usage

```
subset_snp(df, snp.retain, col.snp = SNPs)
```

Arguments

`df`	Data frame containing SNPs.
`snp.retain`	Character vector. Vector specifying which SNPs to keep. SNPs in `snp.retain` must match SNPs in `col.snp` in `df`.
`col.snp`	Symbol. Column containing SNPs.

Details

Subset data frame for specific SNPs only.

Value

Data frame containing only specified SNPs. If no SNP in `snp.retain` matches a SNP in `col.snp`, `subset_snp()` stops with a warning saying *"No SNP in 'snp.retain' matches a SNP in 'col.snp'. Could not filter for SNP."*.

See Also

`extract_snp()`, `count_snp()`, `get_snp()`

Other subset functions: `subset_df()`, `subset_mir_threshold()`, `subset_mir()`, `subset_research()`, `subset_review()`, `subset_year()`

`subset_year` *Subset data frame for abstracts published in a specific period*

Description

Subset data frame for abstracts published in a specific period only.

Usage

```
subset_year(df, col.year = Year, start = NULL, end = NULL)
```

Arguments

`df`	Data frame containing publication years.
`col.year`	Symbol. Column containing publication years.
`start`	Integer. Optional. Beginning of publication period. If `start = NULL`, `start` is set to the least recent year in `df`.
`end`	Integer. Optional. End of publication period. If `end = NULL`, `end` is set to the most recent year in `df`.

Details

Subset data frame for abstracts published in a specific period only. All other abstracts published not within this period are silently dropped.

Value

Data frame containing abstracts published in a specific period only.

See Also

`subset_research()`, `subset_review()`

Other subset functions: `subset_df()`, `subset_mir_threshold()`, `subset_mir()`, `subset_research()`, `subset_review()`, `subset_snp()`

Index