Package 'bioTM'

August 9, 2023

Type Package

Title The bio-text miner

Version 0.0.1

Author Christian A. Hernandez-Fajardo

Maintainer Herandez-Fajardo, C.A <christian.a.hernandezf@hotmail.com>

Description Type a query and data mine hundreds of related PubMed articles. Results are highly relevant keywords or bioconcept relationships that can be viewed or exported.

License

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

abstractReduction 1
add_underscores_to_compound_words 2
AOP_XML_children_organizer 2
bioCAF 3
easyKAF 3
europepmc_retrieval 4
KAFviewer 4
matchAOP 5
matchMeSH 5
MeSH_filter 6
MeSH_Mapper 6
pubmed_retrieval 7
pubMine 7
pubParse 8
pubRetrieve 8
PubTator 9
rm_negation 9
Rule_Concatenator 10
Word_Cleaner 10

abstractReduction *Reduce the size of your PubMed abstract texts*

Usage

```
abstractReduction(data, fraction)
```

Arguments

data	PubMed entry data (see 'pubRetrieve')
fraction	The decimal represenitng the fraction of each abstract to trim. We recommend (0.2) or 20 Trim abstracts from the head and remove any entries without abstract text data. abstractReduction(data = abstract_data, fraction = 0.2) Text reduction

add_underscores_to_compound_words *Replace bioentity noun phrase spacing with underscores to preserve structure*

Description

This function is not intended for users

Usage

```
add_underscores_to_compound_words(character_list)
```

Arguments

character_list	List of named entities in character type

AOP_XML_children_organizer *Converts AOP-wiki XML files into R data frames for further processing*

Description

This function takes XML and converts

Usage

```
AOP_XML_children_organizer(doc)
```

Arguments

doc	XML file downloaded from AOP-wiki

Examples

```
Word_Cleaner(data = association_rules_df)
```

`bioCAF` *Mine bioconcept associations by combining PubTator and KAF*

Description

Bioconcepts associations this function can produce are chemical-gene, gene-gene, or disease-chemical/disease-gene.

Usage

```
bioCAF(data, choice, support, venv_path, lang_model)
```

Arguments

`data`	Generated using PubTator
`choice`	Control which associations to mine and retrieve for analysis: "All", Chemical", "Disease", "Gene", "Systox" (Chemical and disease associations)
`support`	To control computational load, support is modifiable according to each query and user needs. If you are generating too many results for your liking, you can increase this support threshold which represents the percentage of (=<2000) abstracts
`venv_path`	REQUIRED: The path to your python venv. (see 'https://github.com/dandycodingpipe/KAFtool' for additional information)
`lang_model`	REQUIRED: The spaCy language model installed in your venv

Examples

```
bioCAF(data, "Disease", 0.001, "C:/Users/Chris/OneDrive/2023/Systox/venvJune19", "en_core_web_lg")
```

`easyKAF` *Quickly mine PubMed literature from either PubMed or Europe PMC for signficant associations*

Description

This method is for general literature exploration and limits users control for quick results. For more control over parameters, see 'pubRetrieve', 'pubParse', and 'pubMine'. For generating results that may be useful for translational research see 'bioKAF'. If this function does not work, you probably defined the venv wrong. You will need to restart R, and retry with the correct path.

Usage

```
easyKAF(query, database, venv_path, lang_model)
```

Arguments

query	Your typical PubMed query. Optimizing your query using the proper PubMed or Europe PMC syntax improves results.
database	Define the PubMed database to retrieve from: "pubmed" or "pmc". Currently, only "pubmed" articles can be used for bio-entity mining.
venv_path	REQUIRED: The path to your python venv. (see 'https://github.com/dandycodingpipe/KAFtool' for additional information)
lang_model	REQUIRED: The spaCy language model installed in your venv

Examples

```
rules <- easyKAF(venv = "C:/Users/JohnDoe/venv/mar6", lang_model = "en_core_web_sm")
```

---

europepmc_retrieval *The Europe PMC search and retrieve function used for sourcing abstracts.*

Description

This function allows you to query and access key entry data from European PubMed central.

Usage

```
europepmc_retrieval(query, retmax)
```

Arguments

query	Your typical PubMed query. Optimizing your query using the proper pubmed syntax improves results!
retmax	Define how many entries you'd like to access for a query. Keep it between 500-1000 for fastest results. (Limit is around 20,000)

Examples

```
europepmc_retrieval(Query = "Vape smoking AND toxicity", retmax = 750)
```

KAFviewer *The KAF visualization suite*

Description

This function allows you to display rules as a bargraph or as a dataframe according to the classification of choice.

Usage

```
KAFviewer(rules, viz, class)
```

Arguments

rules	The data frame of rules you want to classify
viz	The visualization option ("df" or "bar")
class	The classification display option ("all","bme","anatomy", "organism","diseases","chemicals","technic

Examples

```
ruleViewer(viz <- ruleViewer(classified_rules, "bar", "bme"))
```

---

matchAOP — *Compare KAF results to the AOP-wiki database to find notable key events, stressors, and biological processes present in your PubMed query.*

Description

(In development) Uses locality-sensitive hashing to efficiently cluster and calculate the Jaccard similarity between mined rules and AOP-wiki values/classes. Values below 60

Usage

```
matchAOP(sample, AOPdownload)
```

Arguments

rules	Association rule dataframe that was mined using Abstract_ARM or easyKAF. Limit this parameter to 5000-20000 rules as deduplication is very computationally demanding.

Examples

```
KAFxAOP(rules)
```

---

matchMeSH — *The KAF classification work-horse*

Description

This function handles the classification suite of functions and outputs cleaned and classified association rules

Usage

```
matchMeSH(raw_rules, removal)
```

Arguments

raw_rules	The data frame of rules you want to classify
removal	A list of words that crash the classifier. These words are removed and not classified.

Examples

```
matchMeSH(rules, removal = c("ml(-1","sub>2</sub","study","lead","±", "°", "confidence", "-", "%", "", ">", "s
```

`MeSH_filter` *This function simply rebuilds the association rules that the MeSH cleaner had removed.*

Description

Used for MeSH_Cleaner.

Usage

```
MeSH_filter(MeSH_data)
```

Arguments

`MeSH_data`	The data frame of rules you want to re-structure

Examples

```
MeSH_filter(MeSH_data)
```

`MeSH_Mapper` *A function that communicates with the Medical Subject Headings (MeSH) RDF API to classify consequenets (RHS) from mined association rules.*

Description

This function is the classification workhouse which outputs the same association-rule dataframe you input, but with an additional column dedicated to the classification values.

Usage

```
MeSH_Mapper(word, removal)
```

Arguments

`word`	The data frame of rules you want to classify
`removal`	A list of words you'd like to remove from the classification task (typically because they crash the process)

Examples

```
MeSH_Mapper(rules, removal = c("ml(-1","sub>2</sub","study","lead","±", "°"))
```

pubmed_retrieval *PubMed multi-entry and retrieval function*

Description

This function allows you to query and access key entry data from PubMed.

Usage

```
pubmed_retrieval(query, retmax)
```

Arguments

query	Your typical PubMed query. Optimizing your query using the proper pubmed syntax improves results!
retmax	Define how many entries you'd like to access for a query. Keep it between 500-1000 for fastest results. (Limit is around 2000)

Examples

```
pubmed_retrieval(Query = "Vape smoking AND toxicity", retmax = 750)
```

pubMine *Mine association-rules on parsed PubMed abstract data*

Description

The output is an R dataframe containing association rules and associated metrics. Outputs can vary greatly depending on the parsing method.

Usage

```
pubMine(data, min_supp, min_conf, min_p)
```

Arguments

data	Parsed data (See pubParse)
min_supp	Define the minimum support threshold for the rules (e.g 0.01 or 0.10)
min_conf	Define the minimum confidence threshold for the rules (e.g 0.50 or 0.90)
min_p	Define the minimum p-value threshold for the rules (e.g 0.05 or 0.0.005)

Examples

```
pubMine(data <- data_from_Text_Parser, min_supp = 0.01, min_conf = 0.75, min_p = 0.005)
```

pubParse *Parse retrieved abstract data using spaCy*

Usage

```
pubParse(data, method, composite, venv_path, lang_model, reduced_search)
```

Arguments

data	PubMed entry data (see 'pubRetrieve')
method	Filtering method ('POS' or 'DEP'). Default: 'POS'
composite	'Y' or defaults to null. Construct composite words using n-grams. This may help reduce trivial associations between words that form a composite word (e.g insulin -> resistance)
venv_path	REQUIRED: The path to your python venv. (see 'https://github.com/dandycodingpipe/KAFtool' for additional information)
lang_model	REQUIRED: The spaCy language model installed in your venv
reduced_search	The decimal represenitng the fraction of each abstract to trim. We recommend (0.2) or 20
	Parsed PubMed abstracts and extract potentially relevant information according to simple Parts-of-Speech or Language dependencies. For bio-entity extraction see '
	pubParse(data = abstract_data, venv_path = "C:/Users/Chris/OneDrive/2023/Systox/venvJune19", lang_model = "en_core_web_lg", reduced_search = 0.2)
	Text dependencies, language lemmatization, n-grams of parsing, parts segmentation, sentence speech, tokenization,

pubRetrieve *Extract PubMed or Europe PMC entry information for any query*

Description

Combines functionality from two R-packages: "easyPubMed" and "europepmc" for reliable PubMed entry data extraction.

Usage

```
pubRetrieve(query, size, database)
```

Arguments

query	Your typical PubMed query. Optimizing your query using the proper PubMed or Europe PMC syntax improves results.
size	An integer representing the number of entries to retrieve. PubMed is limited to about 2000 articles, while Europe PMC can pull around 20,000.
database	Define the PubMed database to retrieve from: "pubmed" or "pmc".

Examples

```
pubRetrieve(query = "Vape smoking AND toxicity", size = 2000, database = "pubmed")
```

PubTator *Retrieve bio-entities in PubMed data using PubTator*

Description

This function uses PubTator-annotated articles to retrieve pre-annotated articles. These results should be passed onto bioKAF for further processing (See bioKAF).

Usage

```
PubTator(query)
```

Arguments

query	Input a PubMed query (see https://pubmed.ncbi.nlm.nih.gov/help/#how-do-i-search-pubmed)

Examples

```
PubTator("dioxin toxicity","C:/Users/Chris/OneDrive/2023/Systox/venvJune19", "en_core_web_lg" )
```

rm_negation *The function removes sentences possessing negations. Used in Text_Parser()*

Description

This function removes negations in already-parsed abstract corpus.

Usage

```
rm_negation(Abstract_Parse)
```

Arguments

Abstract_Parse	Input your parsed abstracts. (typically obtained from the spacy_parse function)

Examples

```
rm_negation(Abstract_Parse = parsed_abstracts_from_spacy_parse)
```

Rule_Concatenator *Clean association rules*

Description

Before fuzzy matching we have to pre-process the rules by concatenating them into a single string

Usage

```
Rule_Concatenator(rules)
```

Arguments

rules	The data frame of rules you want to concatenate

Examples

```
Rule_Concatenator(rules)
```

Word_Cleaner *A crude consequent or RHS remover function for ambigious rules.*

Description

This function removes the bracketing present in association-rules (e.g consequent) for better post-processing

Arguments

data	Input your mined associations rules.

Examples

```
Word_Cleaner(data = association_rules_df)
```