Package 'petro.One'

January 13, 2019

Type Package

Title Statistics and Text Mining for Oil and Gas Papers from OnePetro Metadata

Version 0.2.3

Description Application that retrieves papers metadata from the OnePetro website. Thousands of papers on oil and gas live in OnePetro. By retrieving metadata from the search queries, a summary of papers that match the query words, can be retrieved for further analysis and text mining. There are some statistics and data mining provided such as word cloud plots, keywords frequency, conversion to corpus document, and removal of common usage words. OnePetro link: <https://www.onepetro.org/>.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests testthat, mockery, knitr, rmarkdown, covr

Imports rvest, xml2, magrittr, tibble, tm, wordcloud, urltools, dplyr, graph, Rgraphviz, ggplot2, RWeka, cluster, RColorBrewer, data.table, SnowballC

VignetteBuilder knitr

URL https://github.com/f0nzie/petro.One

NeedsCompilation no

Author Alfonso R. Reyes [aut, cre]

Maintainer Alfonso R. Reyes <alfonso.reyes@oilgainsanalytics.com>

Repository CRAN

Date/Publication 2019-01-13 16:20:03 UTC

R topics documented:

`petro.One-package` *Text mining and statistics for OnePetro papers petro.One*

Description

Text mining and statistics for OnePetro papers petro.One

custom_stopwords *Default custom stop words*

Description

This is a minimal dataset of custom stopwords. You can supply your own stopwords by editing the file stopwords.txt under ‘extdata‘ and then importing it. The provided dataset is a basic way to start and eliminate common words from the paper titles during classification.

Dataset: stopwords.rda

Source: stopwords.txt

Usage

```
custom_stopwords
```

Format

An object of class `NULL` of length 0.

discipline_labels *Discipline and Subject labels dataset*

Description

dataset containing disciplines and subjects. The purpose is to categorize papers based on the words in their title since OnePetro does not supply keywords or any sort of categorization. File: disciplines.rda Class: data.frame

Usage

```
discipline_labels
```

Format

An object of class `data.frame` with 254 rows and 2 columns.

generate_offline_data *Generate data for offline testing Mockup test data*

Description

Generate data for offline testing Mockup test data

Usage

```
generate_offline_data()
```

get_papers_count *Number of paper for a given query*

Description

Obtains the number of papers being queried by the URL

Usage

```
get_papers_count(url)
```

Arguments

url	char a query URL for OnePetro

Examples

```
## Not run:
# Example 1
url_1 <- make_search_url(query = "static gradient survey", how = "all")
get_papers_count(url_1)
#
# Example 2
url_2 <- make_search_url(query = "vertical lift performance", how = "all")
get_papers_count(url_2)
#
# Example 3
url_3 <- make_search_url(query = "inflow performance relationship", how = "all")
get_papers_count(url_3)

## End(Not run)
```

get_term_document_matrix *A TermDocumentMatrix corpus objects*

Description

Transforms a document into a VCorpus TermDocumentMatrix object plus additional calculated matrix, row sums and words frequency objects

Usage

```
get_term_document_matrix(df)
```

Arguments

df	a dataframe with paper results

Value

a list

---

get_top_term_papers *Get papers for top "N" terms*

Description

Indicate the top terms from which we want to extract papers. For instance, if we want the papers for the top 10 terms, we set top_terms = 10.

Usage

```
get_top_term_papers(papers, tdm_matrix, top_terms, terms = NULL,
  verbose = FALSE)
```

Arguments

papers	a dataframe with papers
tdm_matrix	a Term Document Matrix
top_terms	top 10, or top 20, etc.
terms	a term or vector of terms to get papers from
verbose	set to TRUE to show progress

---

join_keywords *Get paper count and paper dataframe by joining keywords as vectors*

Description

Get paper count and paper dataframe by joining keywords as vectors

Usage

```
join_keywords(..., get_papers = TRUE, bool_op = "AND", sleep = 3,
  verbose = FALSE)
```

Arguments

...	input character vectors
get_papers	generate or not a dataframe with papers
bool_op	boolean operator. It can be AND or OR
sleep	seconds to wait before a new quiery to OnePetro
verbose	show progress if TRUE

Examples

```
## Not run:
major  <- c("water-injection", "water injection")
minor  <- c("machine-learning", "machine learning")
lesser <- c("algorithm")
p.df <- join_keywords(major, minor, lesser, get_papers = TRUE,
                      sleep = 2, verbose = FALSE)

## End(Not run)
```

---

make_search_url *Make a search URL for OnePetro*

Description

Create a URL that works in OnePetro website

Usage

```
make_search_url(query = NULL, start = NULL, from_year = NULL,
  peer_reviewed = NULL, published_between = NULL, rows = NULL,
  to_year = NULL, dc_type = NULL, how = "any")
```

Arguments

query	char any words that will be searched
start	int optional to set the starting paper
from_year	int optional to indicate starting year
peer_reviewed	logical optional, TRUE or FALSE
published_between	logical automatic if from_year or to_year are on
rows	int optional. number of papers to retrieve. max=1000
to_year	int optional to indicate end year
dc_type	char optional to indicate if journal, conference paper
how	char default="any". "all" will match exact words

Examples

```
## Not run:
# Example 1
url_1 <- make_search_url(query = "flowing gradient survey", how = "all")
onepetro_page_to_dataframe(url_1)
# Example 2
url_2 <- make_search_url(query = "static  gradient survey", how = "all")
onepetro_page_to_dataframe(url_2)
# Example 3
url_3 <- make_search_url(query = "downhole flowrate measurement",
      how = "all", from_year = 1982, to_year = 2017)
onepetro_page_to_dataframe(url_3)

## End(Not run)
```

`onepetro_page_to_dataframe` *Reads a OnePetro URL and converts it to a dataframe*

Description

A OnePetro URL with a query is read into a HTML page and converted to a dataframe

Usage

```
onepetro_page_to_dataframe(url)
```

Arguments

`url`	char a OnePetro type URL

Examples

```
## Not run:
# Example 1
# Search papers with keyword "smartwell"
url_sw <- "https://www.onepetro.org/search?q=smartwell"
onepetro_page_to_dataframe(url_sw)
# Example 2
# Search for exact words ""vertical lift performance"
url_vlp <- "https://www.onepetro.org/search?q=%22vertical+lift+performance%22"
onepetro_page_to_dataframe(url_vlp)

## End(Not run)
```

papers_by_publication *Papers by publication*

Description

Generate a summary by publications. These publications could be World Petroleum Congress, Annual Technical Meeting, SPE Unconventional Reservoirs Conference, etc.

Usage

```
papers_by_publication(url)
```

Arguments

url	a OnePetro query URL

Examples

```
## Not run:
# Example
my_url <- make_search_url(query = "industrial drilling", how = "all")
papers_by_publication(my_url)

## End(Not run)
```

papers_by_publisher *Papers by publisher*

Description

Generate a summary by publisher. Know publishers: OTC, SPE, etc.

Usage

```
papers_by_publisher(url)
```

Arguments

url	a OnePetro query URL

Examples

```
## Not run:
# Example
my_url <- make_search_url(query = "shale gas", how = "all")
papers_by_publisher(my_url)

## End(Not run)
```

papers_by_type *Summary by document type*

Description

Generate a summary by document type. Types are: conference-paper, journal-paper, presentation, media, other, etc.

Usage

```
papers_by_type(url)
```

Arguments

url	a OnePetro page with results

Examples

```
## Not run:
#
# Example 1
my_url <- make_search_url(query = "well test", how = "all")
papers_by_type(my_url)

## End(Not run)
```

papers_by_year *Papers by Year*

Description

Generate a summary by the year the paper was published

Usage

```
papers_by_year(url)
```

Arguments

url	a OnePetro query URL

Examples

```
## Not run:
# Example
my_url <- make_search_url(query = "production automation", how = "all")
papers_by_year(my_url)

## End(Not run)
```

plot_bars *Plot frequency distribution with horizontal bara*

Description

SHows a bar plot with words on the y-axis and frequency on the x-axis

Usage

```
plot_bars(df, gram.min = 1, gram.max = 1, min.freq = 25)
```

Arguments

df	a dataframe with paper results
gram.min	minimum number of grams
gram.max	maximum number of grams
min.freq	minimum frequency of the words to be plotted

Examples

```
## Not run:
my_url <- make_search_url(query = "well test",
                          dc_type = "conference-paper",
                          from_year = 2017,
                          to_year   = 2018,
                          how = "all")
df <- read_multidoc(my_url)   # create a dataframe of papers
(tf <- term_frequency(df))    # create a term frequency table
min_freq <- min(head(tf, 20)$freq)
plot_bars(df, min.freq = min_freq)

## End(Not run)
```

plot_cluster_dendrogram *Plot a dendrogram*

Description

Plots a clustering diagram of terms

Usage

```
plot_cluster_dendrogram(df)
```

Arguments

df	a dataframe with paper results

plot_relationships *Plot a relationship diagram with weights*

Description

Plots a diagram with relationships between words. The lines that link the terms are weighted according to how often the connect together

Usage

```
plot_relationships(df, ..., min.freq = 25, threshold = 0.1)
```

Arguments

df	a dataframe with paper results
...	additional parameters
min.freq	minimum frequency of the words to be plotted
threshold	correlation threshold

Examples

```
## Not run:
my_url <- make_search_url(query = "well test",
                          dc_type = "conference-paper",
                          from_year = 2017,
                          to_year   = 2018,
                          how = "all")
df <- read_multidoc(my_url)   # create a dataframe of papers
(tf <- term_frequency(df))    # create a term frequency table
min_freq <- min(head(tf, 20)$freq)
plot_relationships(df, min.freq = min_freq, threshold = 0.075)

## End(Not run)
```

plot_wordcloud *Plot a word cloud*

Description

Plots a cloud plot of words where the size of the words is determined by their frequency

Usage

```
plot_wordcloud(df, ..., max.words = 200, min.freq = 50)
```

Arguments

df	A dataframe with paper results
...	other parameters
max.words	the maximum words to process
min.freq	the minimum frequency of words allowed

---

read_multidoc *Read all OnePetro papers metadata by type of document*

Description

Function iterates through all found document types and extracts papers into a common dataframe

Usage

```
read_multidoc(my_url)
```

Arguments

my_url	OnePetro query URL

---

read_multipage *Reads metadata in groups of 1000 papers*

Description

This function will loop over and grab data from the OnePetro results in groups of 1000 papers at a time. OnePetro limits the number of papers to view to 1000 papers and the query in this function automatically sets the start counter to read them in groups.

Usage

```
read_multipage(url, verbose = FALSE)
```

Arguments

url	A OnePetro query URL
verbose	indicate if want more printing

read_onepetro *Read OnePetro web page given a query URL*

Description

Read a OnePetro webpage using a query URL. Uses xml2 function read_html

Usage

```
read_onepetro(url)
```

Arguments

url	char a query URL for OnePetro

remove_duplicates_by *Remove duplicate papers by a variable*

Description

Duplicates are removed in a dataframe containing papers

Usage

```
remove_duplicates_by(df, by = "book_title")
```

Arguments

df	dataframe of papers
by	variable

Examples

```
## Not run:
major <- c("data driven")
minor <- c("drilling")
dd_drilling <- join_keywords(major, minor, get_papers = TRUE, sleep = 3,
                             verbose = FALSE)
remove_duplicates_by(dd_drilling$papers, by ="paper_id" )

## End(Not run)
```

run_papers_search *Run a papers search providing multiple keywords and optionally save results.*

Description

This search performs search of papers by provifing multiple levels of keywords. The levels can have one or more keywords and the levels can be as many as desired. Deeper levels makes the search longer.

Usage

```
run_papers_search(..., get_papers = TRUE, sleep = 3, verbose = TRUE,
  len_keywords = 3, allow_duplicates = TRUE, save_to_rda = FALSE)
```

Arguments

...	keywords and keyword levels
get_papers	TRUE to retrieve the papers. FALSE, only return the count
sleep	delay in seconds between search to OnePetro
verbose	TRUE if we want internal messages of the search progress
len_keywords	length of the keywords to form the filename of the rda file
allow_duplicates	if TRUE, it will allow duplicates based on book_title and paper_id
save_to_rda	logical that indicates if we want to save results to an RDA

Examples

```
## Not run:
major <- c("gas influx")
minor <- c("overbalanced", "shut in")
lesser <- c("shale", "drilling")
more <- c("gas diffusion", "concentration gradient")
paper_results <- run_papers_search(major, minor, lesser, more,
                                   get_papers = TRUE,       # return with papers
                                   verbose = FALSE,         # show progress
                                   len_keywords = 4,        # naming the data file
                                   allow_duplicates = FALSE) # by paper title and id

## End(Not run)
```

summary_by_dates *Summary by year*

Description

Generate a summary by the year the paper was published

Usage

```
summary_by_dates(result)
```

Arguments

result	a OnePetro page with results

Examples

```
## Not run:
# Example
my_url <- make_search_url(query = "production automation", how = "all")
result <- read_onepetro(my_url)
summary_by_dates(result)

## End(Not run)
```

summary_by_doctype *Summary by document type*

Description

Generate a summary by document type. Types are: conference-paper, journal-paper, presentation, media, other, etc.

Usage

```
summary_by_doctype(result)
```

Arguments

result	a OnePetro page with results

Examples

```
## Not run:
# Example 1
my_url <- make_search_url(query = "well test", how = "all")
result <- read_onepetro(my_url)
summary_by_doctype(result)

## End(Not run)
```

---

summary_by_publications *Summary by publication*

Description

Generate a summary by publications. These publications could be World Petroleum Congress, Annual Technical Meeting, SPE Unconventional Reservoirs Conference, etc.

Usage

```
summary_by_publications(result)
```

Arguments

result	a OnePetro page with results

Examples

```
## Not run:
# Example
my_url <- make_search_url(query = "industrial drilling", how = "all")
result <- read_onepetro(my_url)
summary_by_publications(result)

## End(Not run)
```

---

summary_by_publisher *Summary by publisher*

Description

Generate a summary by publisher. Know publishers: OTC, SPE, etc.

Usage

```
summary_by_publisher(result)
```

Arguments

result	a OnePetro page with results

Examples

```
## Not run:
# Example
my_url <- make_search_url(query = "shale gas", how = "all")
page <- read_onepetro(my_url)
summary_by_publisher(page)

## End(Not run)
```

---

term_frequency *Word Frequency Dataframe*

Description

Returns a dataframe of words vs frequency

Usage

```
term_frequency(df, gram.min = 1, gram.max = 1)
```

Arguments

df	a dataframe with paper results
gram.min	minimum number of grams
gram.max	maximum number of grams

Examples

```
## Not run:
my_url <- make_search_url(query = "neural network",
                          from_year = 1990,
                          to_year   = 1999,
                          how = "all")
df <- onepetro_page_to_dataframe(my_url)
term_frequency(df)

## End(Not run)
```

term_frequency_n_grams *Find the frequency for two or more words together*

Description

Use this function when trying to find frequency of two or more words

Usage

```
term_frequency_n_grams(df, gram.min = 2, gram.max = 2, mc.cores = 2,
  stemming = TRUE, more_stopwords = NULL)
```

Arguments

df	a dataframe with paper results
gram.min	minimum amount of words together
gram.max	maximum amount of words together
mc.cores	number of cores
stemming	apply stemming by default
more_stopwords	a vector of additional stop words

use_example *Unpack an example*

Description

Examples are zipped to save disk space and prevent R complaining while creating the package

Usage

```
use_example(which_one = NULL)
```

Arguments

which_one	example number to use

Examples

```
use_example(1)
```

Index