Package 'monkeylearn'

April 13, 2018

Type Package

Title Accesses the Monkeylearn API for Text Classifiers and Extractors

Version 0.2.0

Description Allows using some services of Monkeylearn <http://monkeylearn.com/> which is a Machine Learning platform on the cloud for text analysis (classification and extraction).

License GPL (>= 2)

LazyData TRUE

URL http://github.com/ropensci/monkeylearn, http://ropensci.github.io/monkeylearn/

BugReports http://github.com/ropensci/monkeylearn/issues

Encoding UTF-8

RoxygenNote 6.0.1.9000

Suggests knitr, rmarkdown, testthat

Imports cowsay, digest, dplyr, httr, jsonlite, magrittr, methods, purrr, tibble (>= 1.2), tidyr, ratelimitr

VignetteBuilder knitr

NeedsCompilation no

Author Maëlle Salmon [aut, cre] (<https://orcid.org/0000-0002-2815-0399>), Amanda Dobbyn [aut], Thomas Leeper [rev] (Thomas Leeper reviewed the package for rOpenSci, see https://github.com/ropensci/onboarding/issues/45), Jeroen Ooms [ctb], rOpenSci [fnd] (https://ropensci.org/), Earlybird Software [fnd] (http://earlybird.co/)

Maintainer Maëlle Salmon <maelle.salmon@yahoo.se>

Repository CRAN

Date/Publication 2018-04-13 18:39:19 UTC

R topics documented:

`monkeylearn_classifiers` *monkeylearn_classifiers*

Description

List of Monkeylearn classifiers modules

Usage

```
monkeylearn_classifiers(private = FALSE, key = monkeylearn_key(quiet =
  TRUE))
```

Arguments

`private`	default is FALSE, whether to show private modules only instead of private and public modules
`key`	The API key

Details

If you don't have any private modules, `monkeylearn_classifiers(private = TRUE)` returns an empty data.frame.

Value

A data.frame (tibble) with details about the classifiers including their classifier_id which should be used in `monkeylearn_classify`.

Examples

```
## Not run:
monkeylearn_classifiers(private = FALSE)
monkeylearn_classifiers(private = TRUE)

## End(Not run)
```

`monkeylearn_classify` *monkeylearn_classify*

Description

Access to Monkeylearn classifiers modules

Usage

```
monkeylearn_classify(request, key = monkeylearn_key(quiet = TRUE),
  classifier_id = "cl_oFKL5wft", texts_per_req = 200, verbose = TRUE,
  params = NULL)
```

Arguments

`request`	A vector of characters (each text smaller than 50kB)
`key`	The API key
`classifier_id`	The ID of the classifier
`texts_per_req`	Number of texts to be fed through per request (max 200). Does not affect output, but may affect speed of processing.
`verbose`	Whether to output messages about batch requests
`params`	Parameters for the module as a named list. See the second example.

Details

Find IDs of classifiers using https://app.monkeylearn.com/main/explore.

You can use batch to send up to 200 texts to be analyzed within the API (classification or extraction) with each request. So for example, if you need to analyze 6000 tweets, instead of doing 6000 requests to the API, you can use batch to send 30 requests, each request with 200 tweets. The function automatically makes these batch calls and waits if there is a throttle limit error, but you might want to control the process yourself using several calls to the function.

You can check the number of calls you can still make in the API using `attr(output, "headers")$x.query.limit.remaini` and `attr(output, "headers")$x.query.limit.limit`.

Value

A data.frame (tibble) with the results whose attribute is a data.frame (tibble) "headers" including the number of remaining queries as "x.query.limit.remaining". Both data.frames include a column with the (list of) md5 checksum(s) of the corresponding text(s) computed using the `digest digest` function.

Examples

```
## Not run:
text1 <- "my dog is an avid rice eater"
text2 <- "i want to buy an iphone"
request <- c(text1, text2)
output <- monkeylearn_classify(request)
output
attr(output, "headers")
## End(Not run)
```

`monkeylearn_extract` *monkeylearn_extract*

Description

Access to Monkeylearn extractors modules

Usage

```
monkeylearn_extract(request, key = monkeylearn_key(quiet = TRUE),
  extractor_id = "ex_isnnZRbS", texts_per_req = 200, verbose = TRUE,
  params = NULL)
```

Arguments

`request`	A vector of characters (each text smaller than 50kB)
`key`	The API key
`extractor_id`	The ID of the extractor
`texts_per_req`	Number of texts to be fed through per request (max 200). Does not affect output, but may affect speed of processing.
`verbose`	Whether to output messages about batch requests
`params`	Parameters for the module as a named list. See the second example.

Details

Find IDs of extractors using https://app.monkeylearn.com/main/explore. Within the free plan, you can make up to 20 requests per minute.

You can use batch to send up to 200 texts to be analyzed within the API (classification or extraction) with each request. So for example, if you need to analyze 6000 tweets, instead of doing 6000 requests to the API, you can use batch to send 30 requests, each request with 200 tweets. The function automatically makes these batch calls and waits if there is a throttle limit error, but you might want to control the process yourself using several calls to the function.

You can check the number of calls you can still make in the API using `attr(output, "headers")$x.query.limit.remaini` and `attr(output, "headers")$x.query.limit.limit`.

Value

A data.frame with the results whose attribute is a data.frame (tibble) "headers" including the number of remaining queries as "x.query.limit.remaining". Both data.frames include a column with the (list of) md5 checksum(s) of the corresponding text(s) computed using the `digest digest` function.

Examples

```
## Not run:
text <- "In the 19th century, the major European powers had gone to great lengths
to maintain a balance of power throughout Europe, resulting in the existence of
 a complex network of political and military alliances throughout the continent by 1900.[7]
  These had started in 1815, with the Holy Alliance between Prussia, Russia, and Austria.
  Then, in October 1873, German Chancellor Otto von Bismarck negotiated the League of
   the Three Emperors (German: Dreikaiserbund) between the monarchs of Austria-Hungary,
    Russia and Germany."
output <- monkeylearn_extract(request = text)
output
# example with parameters
text <- "A panel of Goldman Sachs employees spent a recent Tuesday night at the
Columbia University faculty club trying to convince a packed room of potential
recruits that Wall Street, not Silicon Valley, was the place to be for computer
scientists.\n\n The Goldman employees knew they had an uphill battle. They were
fighting against perceptions of Wall Street as boring and regulation-bound and
Silicon Valley as the promised land of flip-flops, beanbag chairs and million-dollar
stock options.\n\n Their argument to the room of technologically inclined students
was that Wall Street was where they could find far more challenging, diverse and,
yes, lucrative jobs working on some of the worlds most difficult technical problems."

output <- monkeylearn_extract(text,
                              extractor_id = "ex_y7BPYzNG",
                              params = list(max_keywords = 3,
                                            use_company_names = 1))
attr(output, "headers")
## End(Not run)
```

`monkey_classify` *Monkeylearn classify from a dataframe column or vector of texts*

Description

Independent classifications for each row of a dataframe using the Monkeylearn classifiers modules

Usage

```
monkey_classify(input, col = NULL, key = monkeylearn_key(quiet = TRUE),
  classifier_id = "cl_oFKL5wft", params = NULL, texts_per_req = NULL,
  unnest = TRUE, .keep_all = TRUE, verbose = TRUE, ...)
```

Arguments

input	A dataframe or vector of texts (each text smaller than 50kB)
col	If input is a dataframe, the unquoted name of the character column containing text to classify
key	The API key
classifier_id	The ID of the classifier
params	Parameters for the module as a named list.
texts_per_req	Number of texts to be processed per requests. Minimum value is the number of texts in input; max is 200, as per [Monkeylearn documentation](docs.monkeylearn.com/article/api-reference/). If NULL, we default to 200, or, if there are fewer than 200 texts, the length of the input.
unnest	Should the output column be unnested?
.keep_all	If input is a dataframe, should non-col columns be retained in the output?
verbose	Whether to output messages about batch requests and progress of processing.
...	Other arguments

Details

Find IDs of classifiers using https://app.monkeylearn.com/main/explore.

This function relates the rows in your original dataframe or elements in your vector to a classification particular to that row. This allows you to know which row of your original dataframe is associated with which classification. Each row of the dataframe is classified separately from all of the others, but the number of classifications a particular input row is assigned may vary (unless you specify a fixed number of outputs in params).

The texts_per_req parameter simply specifies the number of rows to feed the API at a time; it does not lump these together for classification as a group. Varying this parameter does not affect the final output, but does affect speed: one batched request of x texts is faster than x single-text requests: http://help.monkeylearn.com/frequently-asked-questions/queries/can-i-classify-or-extract-more-than-
Even if batched, each text still counts as one query, so batching does not save you on hits to the API. See the [Monkeylearn API docs](docs.monkeylearn.com/article/api-reference/) for more details.

You can check the number of calls you can still make in the API using attr(output, "headers")$x.query.limit.remaini
and attr(output, "headers")$x.query.limit.limit.

Value

A data.frame (tibble) with the cleaned input (empty strings removed) and a new column, nested by default, containing the classification for that particular row. Attribute is a data.frame (tibble) "headers" including the number of remaining queries as "x.query.limit.remaining".

Examples

```
## Not run:
text1 <- "Hauràs de dirigir-te al punt de trobada del grup al que et vulguis unir."
text2 <- "i want to buy an iphone"
text3 <- "Je déteste ne plus avoir de dentifrice."
```

```
text_4 <- "I hate not having any toothpaste."
request_df <- tibble::as_tibble(list(txt = c(text1, text2, text3, text_4)))
monkey_classify(request_df, txt, texts_per_req = 2, unnest = TRUE)
attr(output, "headers")
## End(Not run)
```

`monkey_extract` *Monkeylearn extract from a dataframe column or vector of texts*

Description

Independent extractions for each row of a dataframe using the Monkeylearn extractor modules

Usage

```
monkey_extract(input, col = NULL, key = monkeylearn_key(quiet = TRUE),
  extractor_id = "ex_isnnZRbS", params = NULL, texts_per_req = NULL,
  unnest = TRUE, .keep_all = TRUE, verbose = TRUE, ...)
```

Arguments

`input`	A dataframe or vector of texts (each text smaller than 50kB)
`col`	If input is a dataframe, the unquoted name of the character column containing text to extract from
`key`	The API key
`extractor_id`	The ID of the extractor
`params`	Parameters for the module as a named list.
`texts_per_req`	Number of texts to be processed per requests. Minimum value is the number of texts in input; max is 200, as per [Monkeylearn documentation](docs.monkeylearn.com/article/api-reference/). If NULL, we default to 200, or, if there are fewer than 200 texts, the length of the input.
`unnest`	Should the output column be unnested?
`.keep_all`	If `input` is a dataframe, should non-`col` columns be retained in the output?
`verbose`	Whether to output messages about batch requests and progress of processing.
`...`	Other arguments

Details

Find IDs of extractors using https://app.monkeylearn.com/main/explore.

This function relates the rows in your original dataframe or elements in your vector to an extraction particular to that row. This allows you to know which row of your original dataframe is associated with which extraction. Each row of the dataframe is extracted separately from all of the others, but the number of extractions a particular input row is assigned may vary (unless you specify a fixed number of outputs in `params`).

The texts_per_req parameter simply specifies the number of rows to feed the API at a time; it does not lump these together for extraction as a group. Varying this parameter does not affect the final output, but does affect speed: one batched request of x texts is faster than x single-text requests: http://help.monkeylearn.com/frequently-asked-questions/queries/can-i-classify-or-extract-more-than- Even if batched, each text still counts as one query, so batching does not save you on hits to the API. See the [Monkeylearn API docs](docs.monkeylearn.com/article/api-reference/) for more details.

You can check the number of calls you can still make in the API using attr(output, "headers")$x.query.limit.remaini and attr(output, "headers")$x.query.limit.limit.

Find IDs of extractors using https://app.monkeylearn.com/main/explore. Within the free plan, you can make up to 20 requests per minute.

You can use batch to send up to 200 texts to be analyzed within the API (classification or extraction) with each request. So for example, if you need to analyze 6000 tweets, instead of doing 6000 requests to the API, you can use batch to send 30 requests, each request with 200 tweets. The function automatically makes these batch calls and waits if there is a throttle limit error, but you might want to control the process yourself using several calls to the function.

You can check the number of calls you can still make in the API using attr(output, "headers")$x.query.limit.remaini and attr(output, "headers")$x.query.limit.limit.

Value

A data.frame (tibble) with the cleaned input (empty strings removed) and a new column, nested by default, containing the extraction for that particular row. Attribute is a data.frame (tibble) "headers" including the number of remaining queries as "x.query.limit.remaining".

Examples

```
## Not run:
text <- "In the 19th century, the major European powers had gone to great lengths
to maintain a balance of power throughout Europe, resulting in the existence of
a complex network of political and military alliances throughout the continent by 1900.[7]
These had started in 1815, with the Holy Alliance between Prussia, Russia, and Austria.
Then, in October 1873, German Chancellor Otto von Bismarck negotiated the League of
the Three Emperors (German: Dreikaiserbund) between the monarchs of Austria-Hungary,
Russia and Germany."
output <- monkeylearn_extract(request = text)
output


# Example with parameters
text <- "A panel of Goldman Sachs employees spent a recent Tuesday night at the
Columbia University faculty club trying to convince a packed room of potential
recruits that Wall Street, not Silicon Valley, was the place to be for computer
scientists.\n\n The Goldman employees knew they had an uphill battle. They were
fighting against perceptions of Wall Street as boring and regulation-bound and
Silicon Valley as the promised land of flip-flops, beanbag chairs and million-dollar
stock options.\n\n Their argument to the room of technologically inclined students
was that Wall Street was where they could find far more challenging, diverse and,
yes, lucrative jobs working on some of the worlds most difficult technical problems."

output <- monkey_extract(text,
```

```
                            extractor_id = "ex_y7BPYzNG",
                            params = list(max_keywords = 3,
                            use_company_names = 1))
    attr(output, "headers")
    ## End(Not run)
```

Index