parser

tokenize_text

```
tokenize_text(sentences)
```

Receives a list of sentences, ie. a sequence of words and returns a tokenized version of it.

remove_stopwords

```
remove_stopwords(text)
```

Receives a list of strings and returns a version of it without stopwords.

Input:

```
- text (list): a list of words (strings).
```

Output:

```
- new_text (list): returns a list of words without portuguese stopwords.
```

remove_punctuation

```
remove_punctuation(text)
```

Receives a list of strings and returns a version of it without punctuation.

Input:

```
- text (list): a list of words (strings).
```

Output:

```
- new_text (list): a list of words without punctuation.
```

stemming

```
stemming(text)
```

```

```

Receives a list of strings and returns a list containing the stemmed version of them.

Input:

```
- text (list): a list of words (strings).
```

Output:

```
- new_text (list): list of string containing stemmed words.
```

lowercase

```
lowercase(text)
```

Receives a list of strings and returns a list with the corresponding lowercase version of these strings.

Input:

```
- text (list): a list of words (strings).
```

Output:

```
- new_text (list): a list of strings with only lowercase characters.
```

lemmatization

```
lemmatization(text)
```

Receives a list of strings and returns a list with the lemmatized version of the verbs contained in the input.

Input:

```
- text (list): list of words (strings).
```

Output:

```
- new_text (list): list of words with verbs lemmatized (if there is any).
```

Parser

```
Parser(self, /, *args, **kwargs)
```

A parser to deal extract text from a url's html, tokenize the text and process it though operations such as stopword and punctuation removal, stemming, lemmatization and lowercase.

extract

```
Parser.extract(self, url)
```

Extract text from a url's html using BeautifulSoup Module. The output is a BeautifulSoup object.

Input:

```
- url (str): an url string.
```

Output:

```
-soup (beautifulSoup object): A nested data structure containing the url's hmtl
elements and content.
```

text_from_web

```
Parser.text_from_web(self, url)
```

Extract text from url's html and returns a list of the strings present on it.

Input:

```
- url (str): an url string.
```

Output:

```
-text (list): a list ofstrings extracted from the url's html.
```

crawler

```
Parser.crawler(self, url, limit=10)
```

Receives an url and returns a list of links (another urls) present on its html.

Input:

```
- url (str): an url string.
- limit (int, 10): a int number, referring to the maximum of links to return.
Default value: 10.
```

Output:

```
- links (list): a list of links found on the url's html.
```

parse

```
Parser.parse(self, url)
```

Receives an url and returns a list of strings extracted from its html already preprocessed (see Parse.preprocess for more information).

Input:

```
- url (str): an url string.
```

Output:

```
- a list of strings that were extracted from the given url's html and that were
processed in order to be lowercase, without stopwords and punctuation.
```

preprocess

```
Parser.preprocess(self, text, use_stem=False, use_lemma=False)
```

Receives a list of strings and returns a list os strings in lowercase, with punctuation and stopwords removed. Optionally, the words are also lemmatized and stemmed.

Input:

```
- text (list): a list of words (strings).
```

Optional:

```
- use_stem (boolean, False): True, if it is to perform stemming in the strings.
Default value: False.
- use_lemma (boolean, False): True, if it is to perform lemmatization in the
strings. Default value: False.
```

Output :

```
- text (list): list of words with all strings in the lowercase version and also
with punctuation and stopwords removed. (Optionally, the strings may also be
stemmed and lemmatized).
```