Szymon Janowski 09.06.2018

## Sprawozdanie

NLTK Python – wyszukiwarka

1. Wczytanie plików.

#Wczytanie plików

*from* nltk.corpus *import* reuters

#Inne niezbędne biblioteki:

*from* nltk.stem.porter *import* \*  
*from* nltk.corpus *import* stopwords

1. Preprocessing.
   1. Tokenizacja,

words = reuters.words(file)

* 1. Różne operacje preprocessingu,
     + Stemming
     + Usuwanie stop-words
     + Zamiana dużych liter na małe

*def* preprocessig(*words*):  
 #delete ending  
 stemmer = PorterStemmer()  
 singles = [stemmer.stem(word) *for* word *in words*]  
  
 #to lower all word  
 lower\_single\_words = [word.lower() *for* word *in* singles]  
  
 # delete stop words  
 stop\_words = set(stopwords.words('english'))  
 words\_without\_stop = [word *for* word *in* lower\_single\_words *if not* word *in* stop\_words]  
  
 #removing punctuations  
 clean\_words =

[word *for* word *in* words\_without\_stop *if* word.isalpha() *or* word.isdigit() *or* word.isalnum()]

*return* clean\_words

* 1. Tworzenie słownika.
     + \_\_dic – klucz: indeks, wartość: słowo
     + \_\_re\_dic – klucz: słowo, wartość: indeks

*def* create\_global\_dictionary\_words(*words*, *iterator*):  
 #do global dictionary with all words in corups  
 *for* word *in words*:  
 *if* word *not in* \_\_dic:  
 \_\_dic[word] = *iterator* \_\_re\_dic[*iterator*] = word  
 *iterator* += 1

* + - \_\_freq – klucz: słowo, wartość: ilość powtórzeń słowa

*def* create\_gloal\_dictionary\_freq\_word\_file(*words*, *file*):  
 all\_freq = {}  
 *for* word *in words*:  
 *if* word *not in* all\_freq:  
 all\_freq[word] = 1  
 *else*:  
 all\_freq[word] += 1  
  
 \_\_word\_freq\_vector\_file[*file*] = all\_freq

1. Uzupełnianie stworzonych słowników dla wszystkich plików z korpusu.

*def* do\_dicts\_for\_all\_file():  
 #fill global dict with words and fill global dic freq and fill  
 iterator = 0  
 i = 0  
 *for* category *in* reuters.categories():  
 *for* file *in* reuters.fileids(category):  
 words = reuters.words(file)  
 clean\_words = preprocessing(words)  
 create\_global\_dictionary\_words(clean\_words, iterator)  
 create\_gloal\_dictionary\_freq\_word\_file(clean\_words,file)

1. Tworzenie macierzy dokument / słowo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Słowo1 | Słowo2 | Słowo3 | … |
| Plik1 | 1 | 0 | 3 | … |
| … | … | … | … | … |

*def* create\_document\_word\_freq\_matrix():  
 *for* file, words *in* \_\_word\_freq\_vector\_file.items():  
 vec = []  
 *for* index, w *in* \_\_re\_dic.items():  
 *if* w *in* words:  
 vec.append(words[w])  
 *else*:  
 vec.append(0)  
 \_\_freq\_vector\_file[file] = vec

1. Co dalej? xD