04

Explorando a tokenização

Júlia é da área de TI e trabalha para uma empresa de marketing que lhe propõe um novo desafio: após realizar sua primeira classificação de sentimentos ela deve otimizar os resultados a fim de tornar possível a criação de uma aplicação. Júlia resolve então explorar outros algoritmos de tokenização na biblioteca NLTK.

Ela resolve comparar as funções:

```
from nltk import tokenize
frase = "Os cursos da Alura são ótimos, além de ótimos, têm alunos ótimos!”
token_espaco = tokenize.WhitespaceTokenizer()
token_pontuacao = tokenize.WordPunctTokenizer()

token_1 = token_espaco.tokenizer(frase)
token_2 = token_pontuacao.tokenizer(frase)
```

“Os cursos da Alura são ótimos, além de ótimos, têm alunos ótimos!”

(Sim, essa frase não é tão agradável do ponto de vista gramatical, mas é um ótimo exemplo para verificar a **diferenças entre as tokenizações**).

Ajude Julia a decidir: Qual será a saída das variáveis token_1 e token_2? Qual será o melhor método de tokenização a se usado na análise de sentimentos?.

Selecione uma alternativa

A

[“Os”, “cursos”, “da”, “Alura”, “são”, “ótimos,”, “além”, “de”, “ótimos,”, “têm”, “alunos”, “ótimos!”] - WhitespaceTokenizer();
[“Os”, “cursos”, “da”, “Alura”, “são”, “ótimos”, “,”, “além”, “de”, “ótimos”, “,”, “têm”, “alunos”, “ótimos”, “!”] - WordPunctTokenizer();
Comparando as saídas dos diferentes algoritmos o mais adequado a se escolher é o WhitespaceTokenizer, pois nesse caso existem duas variações da palavra “ótimos” e no segundo método há somente uma variação dessa mesma palavra.

B

[“Os”, “cursos”, “da”, “Alura”, “são”, “ótimos,”, “além”, “de”, “ótimos,”, “têm”, “alunos”, “ótimos!”] - WordPunctTokenizer();
[“Os”, “cursos”, “da”, “Alura”, “são”, “ótimos”, “,”, “além”, “de”, “ótimos”, “,”, “têm”, “alunos”, “ótimos”, “!”] - WhitespaceTokenizer();
Comparando as saídas dos diferentes algoritmos, o mais adequado a se escolher é o WordPunctTokenizer, visto que na primeira opção de método a palavra “ótimos” tem duas variações (“ótimos,”, “ótimos!”) e no segundo método há somente uma variação da mesma palavra.

C

[“Os”, “cursos”, “da”, “Alura”, “são”, “ótimos,”, “além”, “de”, “ótimos,”, “têm”, “alunos”, “ótimos!”] - WhitespaceTokenizer();
[“Os”, “cursos”, “da”, “Alura”, “são”, “ótimos”, “,”, “além”, “de”, “ótimos”, “,”, “têm”, “alunos”, “ótimos”, “!”] - WordPunctTokenizer();
Comparando as saídas dos diferentes algoritmos o mais adequado a se escolher é o WordPunctTokenizer, visto que na primeira opção de método a palavra “ótimos” tem duas variações (“ótimos,”, “ótimos!”) e no segundo método há somente uma variação da mesma palavra.