

Atividade 02 - Processamento de Linguagem Natural

Aluno: Carlos Eduardo Falandes

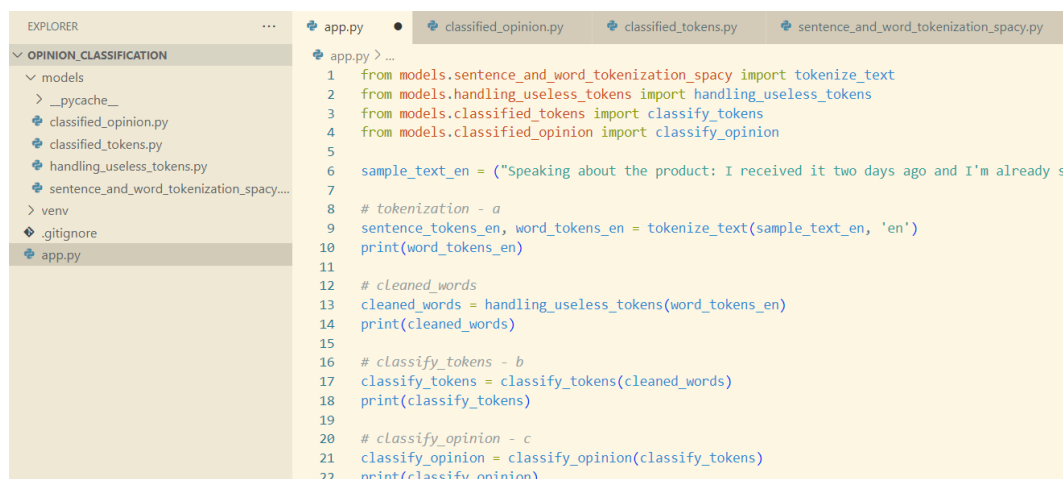
TC.2.2. Qual um possível efeito da não remoção de um iframe ou de scripts e CSS's de um documento capturado através de uma raspagem de dados?

A remoção de marcadores é usada principalmente para atividades de raspagem de dados em rede (web scraping) essa etapa é necessária para não inserir dados ruidosos e não informativos na etapa seguinte da pipeline. Isso ocorre porque iframes e CSS geralmente não contêm o conteúdo informativo, mas sim elementos de apresentação e interatividade, tais elementos podem introduzir ruído nos dados extraídos, isso aumenta a quantidade de ruído nas próximas etapas, já que informações irrelevantes, como anúncios ou dados carregados por scripts e estilos, podem ser incluídas, sem adicionar valor à raspagem.

PP.2.7. Tomando-se como base os problemas TC.2.4 e PP.2.3, exemplifique a classificação de uma opinião (e.g., um comentário sobre um produto na Internet) pela mera busca de um ou mais tokens de palavras. Aplique as etapas de processamento ilustradas na aula de PLN. Na sua resolução mostre o seguinte:

- A obtenção dos tokens a partir da(s) sentença (as) origina(is).
- A identificação dos tokens que caracterizam a opinião positiva, negativa ou neutra.
- A classificação da opinião baseada na mera ocorrência do token (faça um esquema em que um ou mais tokens podem ser utilizados para classificar a opinião).
- Ilustre um caso onde esse mecanismo de classificação não funciona, mesmo que os tokens sejam encontrados.

O algoritmo foi estruturado da seguinte forma: inicialmente, é realizada a tokenização utilizando o algoritmo disponibilizado pelo professor, que eu adaptei para o uso em funções. Em seguida, a lista de tokens resultante é processada para remover elementos irrelevantes, como pontuações e letras maiúsculas. Após essa limpeza, os tokens passam por um classificador que determina se cada um é positivo, negativo ou neutro. Por fim, contamos a frequência de palavras em cada categoria mencionada. Se houver um maior número de tokens positivos, o texto é classificado como positivo; se os negativos forem mais frequentes, o texto é classificado como negativo; e, caso os números sejam iguais, o texto é considerado neutro.



```
1 from models.sentence_and_word_tokenization_spacy import tokenize_text
2 from models.handling_useless_tokens import handling_useless_tokens
3 from models.classified_tokens import classify_tokens
4 from models.classified_opinion import classify_opinion
5
6 sample_text_en = ("Speaking about the product: I received it two days ago and I'm already sa
7
8 # tokenization - a
9 sentence_tokens_en, word_tokens_en = tokenize_text(sample_text_en, 'en')
10 print(word_tokens_en)
11
12 # cleaned_words
13 cleaned_words = handling_useless_tokens(word_tokens_en)
14 print(cleaned_words)
15
16 # classify_tokens - b
17 classify_tokens = classify_tokens(cleaned_words)
18 print(classify_tokens)
19
20 # classify_opinion - c
21 classify_opinion = classify_opinion(classify_tokens)
22 print(classify_opinion)
```

Entretanto, existem limitações na análise de opinião realizada dessa forma, uma vez que o algoritmo não avalia o sentido geral da frase. Isso pode resultar na falta de compreensão de contradições ou qualificadores, como nas seguintes frases: "The product is good, but it has some flaws." Neste caso, "good" é classificado como positivo, enquanto "flaws" é negativo. O algoritmo contaria ambos, resultando em uma classificação neutra. Da mesma forma, na frase "It's nice, but too expensive," a palavra "nice" sugere uma avaliação

positiva, mas "too expensive" apresenta um argumento negativo que o algoritmo ignora. Exemplos como esses ilustram as fragilidades do algoritmo; no entanto, ele ainda funciona de maneira eficaz em uma análise geral.

A linguagem utilizada para desenvolver o algoritmo foi o inglês, devido à facilidade de acesso a bibliotecas que classificam palavras.