NLP

Tokenización

¿Cómo funciona la tokenización?

- La tokenización es un proceso en el que una secuencia se divide en partes como palabras, oraciones, frases, etcétera. La forma en que funcionan estas tokenizaciones se explica a continuación.

Tokenización de palabras

- En este proceso, una secuencia, como una oración o un párrafo, se descompone en palabras. Estas tokenizaciones se llevan a cabo en base al delimitador "espacio" (" "). Digamos que tenemos una frase, "el cielo es azul". Luego, aquí la oración consta de 4 palabras con espacios entre ellas. La tokenización de palabras realiza un seguimiento de estos espacios y devuelve la lista de palabras de la oración. ["el","cielo","es","azul"].

Tokenización de oraciones

- En este proceso, en lugar de tokenizar un párrafo en función del "espacio", lo tokenizamos en función de "." y ",". Por lo tanto, obtenemos todas las diferentes oraciones que componen el párrafo.

Tokenización en Python

- En Python, una forma sencilla de tokenizar un texto es utilizando el método split(), que divide una cadena de texto en palabras basándose en los espacios en blanco. Por ejemplo:

```
texto = "Hola, ¿cómo estás?"
tokens = texto.split()
```

Este código dividirá la cadena texto en una lista de palabras: ['Hola,', '¿cómo', 'estás?'].

Este proceso también se puede llevar a cabo con el módulo NLTK.

¿Qué es NLTK?

- NLTK (Natural Language Toolkit) es una biblioteca de Python muy utilizada para el procesamiento del lenguaje natural (PLN). Proporciona herramientas para tareas como la tokenización, el análisis de sentimientos, el reconocimiento de entidades y mucho más.

- Tokenización con NLTK

NLTK ofrece funciones más avanzadas para la tokenización, como word_tokenize para dividir un texto en palabras y sent_tokenize para dividir un texto en oraciones.

NLTK es muy útil para tareas de PLN más complejas y proporciona una mayor precisión en la tokenización comparado con el método split().

¿Por qué NLTK?

NLTK fue diseñado con cuatro objetivos principales en mente:

- Simplicidad:

Proporcionar un marco intuitivo junto con bloques de construcción sustanciales, brindando a los usuarios un conocimiento práctico de NLP sin empantanarse en el tedioso mantenimiento que generalmente se asocia con el procesamiento de datos de lenguaje anotados.

- Coherencia:

Proporcionar un marco uniforme con interfaces y estructuras de datos consistentes, y nombres de métodos fáciles de adivinar.

- Extensibilidad:

Proporcionar una estructura en la que se puedan acomodar fácilmente nuevos módulos de software, incluidas implementaciones alternativas y enfoques competitivos para la misma tarea.

- Modularidad:

Proporcionar componentes que se puedan usar de forma independiente sin necesidad de comprender el resto del conjunto de herramientas.