

TEMA 2 – HOJA DE EJERCICIOS I

Se proponen diferentes ejercicios de preprocesamiento textual y anotación lingüística aplicados al procesamiento del lenguaje natural. Se proporcionan textos de prueba en inglés y español, aunque se pueden utilizar otros como entrada para realizar las comprobaciones que se consideren necesarias.

Ejercicio 1. Análisis con NLTK

[NLTK](#) es una herramienta popular de código abierto para trabajar con lenguaje natural en Python. Proporciona un acceso rápido y sencillo a decenas de corpus y recursos léxicos, incluyendo diversos conjuntos de herramientas para tareas como tokenización, stemming y etiquetado Part-of-Speech.

Apartado 1.1. Dado el fichero de texto con contenido en inglés *data_science.txt*:

- Implementa un código en Python que divida el texto en frases usando **PunktSentenceTokenizer** de NLTK. Un ejemplo de uso se ve a continuación:

```
import nltk
from nltk.tokenize import PunktSentenceTokenizer

# Download the Punkt package
nltk.download('punkt')

# Sample text
text = "We met Miss. Tanaya Das and Mr.Rohan Singh today. They are pursuing a B.tech degree in Data Science."

# Initialize the PunktSentenceTokenizer
tokenizer = PunktSentenceTokenizer()

# Tokenize the text into sentences
sentences = tokenizer.tokenize(text)

print(sentences)
```

['We met Miss.', 'Tanaya Das and Mr.Rohan Singh today.', 'They are pursuing a B.tech degree in Data Science.']

Tras ello, tokeniza a nivel de palabra. ¿Qué ocurre si se tokeniza utilizando la función `split` de Python en lugar de la función `word_tokenize` de NLTK? Haz pruebas y compara la salida.

- Elimina todos los signos de puntuación que aparezcan en el mismo fichero.
- Identifica las 5 palabras más frecuentes antes y después de eliminar los signos de puntuación. Además, se debe detectar cuántas veces aparece la palabra *data* en el texto. Pista: <https://www.nltk.org/api/nltk.probability.html>

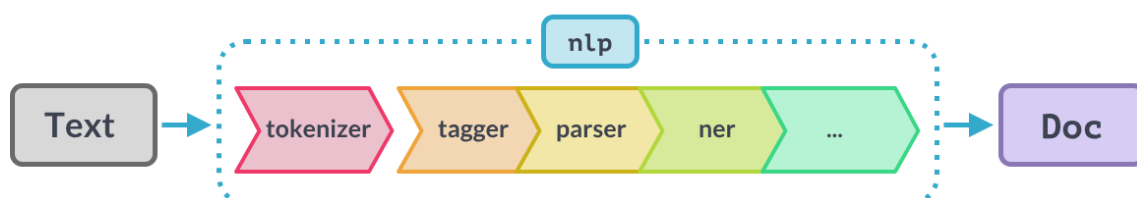
- d. Divide el texto en frases y, por cada una de ellas, elimina las palabras vacías (stopwords) que contengan.
- e. Dada la primera frase del texto, obtén la etiqueta Part-of-Speech para cada token.
- f. Obtén los tokens de la primera frase del texto y, por cada uno de ellos, aplica stemming. Observa la salida, ¿Todos los *stems* obtenidos están en el diccionario?
- g. Obtén tokens de la primera frase del texto y, por cada uno de ellos, obtén su *lema*. ¿En qué se diferencian con los resultados obtenidos en el apartado anterior?
- h. Obtén las entidades nombradas en la totalidad del texto. Pista: https://www.nltk.org/api/nltk.chunk.ne_chunk.html

Apartado 1.2. Repite los anteriores apartados para el texto en español *ciencia_datos.txt*. Compara las diferencias con respecto a los resultados obtenidos en inglés. ¿Funciona bien el etiquetado Part-of-Speech en español? Nota que el lematizador de NLTK por defecto no ofrece soporte para textos en español. Por ello, se debe construir un lematizador propio utilizando un diccionario que asocia distintas formas flexionadas con su lema. Este diccionario se encuentra en *lemmatization_es.txt*.

Ejercicio 2. Análisis con SpaCy

[spaCy](#) es una librería de software de código abierto diseñada para facilitar tareas de procesamiento del lenguaje natural avanzado. Es fácil de usar y destaca por su velocidad a la hora de realizar el procesamiento del texto.

Un pipeline de spaCy consta de una secuencia de componentes que enriquecen con anotaciones un string inicial de texto hasta crear un objeto de tipo Doc:



Es necesario descargar los modelos de los lenguajes que se quieran utilizar. Los modelos disponibles se pueden consultar [aquí](#).

Apartado 2.1. Dado el siguiente texto en español:

En 2026, la tecnología seguirá cambiando de forma notable cómo vivimos y cómo trabajamos cada día. Las tendencias de este año se centran en el impulso de la inteligencia artificial y la automatización, con actores clave como OpenAI y Google DeepMind. Asimismo, destacan una conectividad más potente y soluciones de sostenibilidad basadas en datos.

- a. Muestra para cada token y en una línea diferente, el texto del token, la categoría gramatical y el lema.
- b. Cuenta el número de palabras vacías que se encuentran en el texto.
- c. Muestra los adjetivos y nombres que aparecen en el texto.
- d. Muestra para cada token su categoría gramatical y la explicación asociada a dicha categoría. Para ello se puede utilizar `spacy.explain`, que da una descripción acerca del tag particular del token.
- e. Cuenta el número de frases que hay en el párrafo e imprime cada una de ellas.
- f. Para cada frase, elimina sus stopwords y fórmalas de nuevo sin dichas palabras.
- g. Obtén las entidades nombradas en el texto.
- h. Implementa una función que preprocese y normalice el texto aplicando operaciones habituales: convertirlo a minúsculas, lematizar cada token y eliminar stopwords y signos de puntuación. Se deben agrupar estas operaciones en una única función para poder invocarla cuando sea necesario.

Apartado 2.2. Dado el conjunto de textos en español *open_australia_1.txt* y *open_australia_2.txt*, se quiere extraer, para cada documento, la siguiente información utilizando el matcher de spaCy:

- a. Las cadenas de texto que contengan *Grand Slam*.
- b. Las cadenas de texto que contengan el lema *volver*.
- c. Las cadenas de texto con nombres y adjetivos.
- d. Las cadenas de texto con un nombre propio seguido de un verbo.

Pista: Los atributos válidos para especificar en los patrones y los operadores de cuantificación se pueden encontrar en <https://spacy.io/usage/rule-based-matching>. Estos patrones permiten encontrar chunks útiles para análisis posteriores.

Ejercicio 3. Expresiones regulares

Las expresiones regulares representan una forma fácil y rápida de procesar textos, permitiendo filtrarlo, sustituirlo y encontrar patrones en el mismo. Trabajan directamente sobre la forma del texto sin necesidad de recursos lingüísticos adicionales.

Apartado 3.1. Se pide normalizar y detectar patrones en un texto a partir del uso de expresiones regulares.

Este es un texto que tiene ejemplos de fechas. Hoy es 01/02/2026, esta es una fecha posterior al 13 de enero de 2026. ¿Nos gustaría estar ya a 5 de julio y empezar las vacaciones? Casi que mejor no, que el tiempo avance a su ritmo. El primer día de clase fue el 26-01-2025, y el último día será el 5 de mayo del 2025.”

- a. Obtén todos los tokens dividiendo el texto separando por espacios en blanco.
- b. Elimina los signos de puntuación y las palabras con menos de 3 caracteres.
- c. Detecta las fechas que aparecen (con formatos DD/MM/AAAA, DD-MM-AAAA, DD de MM de AAAA y DD de MM).