# Procesamiento de Lenguaje Natural

**Etapas**

1. **Preprocesamiento/Normalización:**

* Crear uniformidad en los datos.
* Se eliminan stopwords (Palabras que no dan valor como artículos o conectores), por ejemplo “El vaso es rojo” -> “vaso rojo”.
* Se quitan acentos y es mejor apegarse al alfabeto americano.
* Debe quedar un vector de palabras.

1. **Vectorización:**

* Convierto un vector de palabras a uno de números: [palabras] -> [números]

1. **Modelado:**

* Aplica algoritmo de clasificación, sentimiento y/o predicción a los números.

**Conceptos:**

**Token:**  Unidad más chica que puedo interpretar, puede ser una palabra, letra, etc.

**Vocabulario:** Obtener los tokens únicos, para evitar que se repitan.

**Corpus:** Un libro compuesto de documentos o archivos, libro dicho de forma metafórica.

**Documento:** Texto, como una hoja de texto, etc., una hoja del corpus.

**Algoritmos**

**1. Bag of Words (BoW)**

Es un modelo que convierte un conjunto de documentos en vectores. Cada posición del vector representa una palabra del vocabulario, y el valor indica cuántas veces aparece en el documento.

**Ventajas:**

* Simple y rápido.
* Funciona bien en tareas como clasificación.

**Limitaciones:**

* Pierde el orden de las palabras.
* No distingue palabras comunes vs. importantes.

**2. One-Hot Encoding**

Representa cada palabra como un vector binario, donde solo una posición tiene un 1 (la correspondiente a la palabra) y todas las demás son 0.

**Limitaciones:**

* Solo útil para representar palabras individuales.
* Muy ineficiente para vocabularios grandes (vectores dispersos).
* No refleja similitud entre palabras.

**3. TF-IDF (Term Frequency - Inverse Document Frequency)**

Es un modelo que pondera las palabras en un documento, penalizando las que aparecen en muchos documentos y destacando las más relevantes.

**Fórmulas:**

**TF (Frecuencia del término):**  
TF(t, d) = (# apariciones de t en d) / (# términos en d)

**IDF (Frecuencia inversa de documentos):**  
IDF(t) = log( (# documentos) / (1 + # docs que tienen t ) )

**TF-IDF final:**  
TF-IDF(t, d) = TF(t, d) \* IDF(t)

**¿Por qué funciona?**

* Palabras frecuentes en un documento tienen alto TF.
* Palabras comunes en todos los documentos tienen bajo IDF (no aportan discriminación).
* TF-IDF destaca palabras importantes y específicas.