Pre-procesamiento y Exploración de Texto

Andrés Abeliuk, Fabián Villena

Pre-procesamiento de Texto

"80% del tiempo en proyectos de NLP se dedica al preprocesamiento"

Pre-procesamiento del Texto

¿Por qué es necesario?
El texto no estructurado contiene ruido (como caracteres especiales, palabras irrelevantes, o inconsistencias) que dificultan el análisis.

Objetivos del preprocesamiento:

- **Normalización:** Homogeneizar formatos (minúsculas, eliminación de acentos, etc.).
- **Tokenización:** Dividir el texto en palabras, frases o unidades significativas.
- **Eliminación de ruido:** Quitar palabras irrelevantes (stopwords), signos de puntuación o caracteres especiales.
- **Lematización o stemming:** Reducir las palabras a su forma base o raíz para un análisis más coherente.

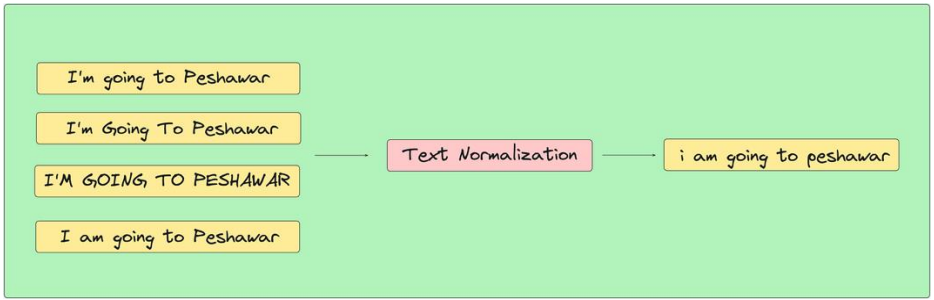
Normalización del texto

La normalización prepara el texto para ser procesado, asegurando coherencia y facilitando el análisis. Es un paso esencial antes de realizar la mayoría de las tareas de Procesamiento de Lenguaje Natural (PLN).

Pasos Principales:

1. **Tokenización (Segmentación de palabras):**
   1. Divide el texto en unidades significativas (tokens), como palabras o subpalabras.
   2. Ejemplo: "El procesamiento de texto" → ["El", "procesamiento", "de", "texto"].

2. **Estandarización de formatos:**
   1. Convierte el texto a un formato uniforme.
      1. Convertir a minúsculas.
      2. Eliminar caracteres especiales o puntuación irrelevante.
      3. Sustituir palabras por variantes estándar (e.g., "décima" → "10ª").



3. **Segmentación de oraciones:**
   1. Identifica los límites entre oraciones.
   2. Ejemplo: "Hola. ¿Cómo estás?" → ["Hola.", "¿Cómo estás?"].

Algoritmo de tokenización

El **algoritmo que se utilice para tokenizar debe ser consistente con el lenguaje que estamos analizando**; distintos lenguajes expresan de distinta manera la utilización de distintos símbolos.

```
>>> text = 'That U.S.A. poster-print costs $12.40...'
>>> pattern = r'''(?x)    # set flag to allow verbose regexps
...     ([A-Z]\.)+        # abbreviations, e.g. U.S.A.
...   | \w+(-\w+)*        # words with optional internal hyphens
...   | \$?\d+(\.\d+)?%?  # currency and percentages, e.g. $12.40, 82%
...   | \.\.\.            # ellipsis
...   | [][.,;"'?():-_`]  # these are separate tokens; includes ], [
... '''
>>> nltk.regexp_tokenize(text, pattern)
['That', 'U.S.A.', 'poster-print', 'costs', '$12.40', '...']
```

Tokenizadores modernos

Los métodos de tokenización modernos se basan en la **distribución de los símbolos** en el conjunto de entrenamiento para construir el vocabulario, permitiendo:

- Dividir el texto no solo en palabras completas, sino también en **subpalabras** o unidades más pequeñas.
- Manejar palabras desconocidas generando combinaciones de subpalabras presentes en el vocabulario.
- Mejor generalización a dominios nuevos

La canción y la actuación, cantando y actuando ando

```
La can# #ción y la actua# #ción , cant# #ando y actu# #ando ando
```

Tokenizadores modernos

Algoritmos Principales:

- BPE (Byte-Pair Encoding): GPT, RoBERTa
- WordPiece: BERT, DistilBERT
- SentencePiece (Google): Independiente del idioma: T5, mBERT

Ex1) "ABABCABCD"
→ XXCXCD : X = AB
→ XYYD : Y = XC

Ex2) "aaabdaaabac"
→ ZabdZabac : Z = aa
→ ZYdZYac : Y = ab
→ XdXac : X = ZY

Byte-pair encoding:

Este algoritmo parte con un vocabulario que consiste en todos los caracteres presentes en el texto e **iterativamente va agregando subpalabras al vocabulario al unir los pares adyacentes** de símbolos más frecuentes.

Tokenizadores y modelos

En el modelo de BERT en español desarrollado por el DCC se utilizó un tokenizador basado en el algoritmo SentencePiece que aprende el vocabulario desde el conjunto de entrenamiento y genera sub-palabras.

La biblioteca `tokenizers` contiene todas las implementaciones de tokenizadores modernos.

https://huggingface.co/dccuchile/bert-base-spanish-wwm-uncased

- ##viste
- tienda
- ##ple
- ##nud
- mó
- rode
- pris
- presentó
- quedó
- ##telo
- ##gados
- lev
- ##arle
- dipu
- europeas
- ##rue

Lematización

Proceso de **reducir palabras a su forma base o "lema"** para normalizar el texto.

- La lematización considera el contexto gramatical.
- Ejemplo: "corriendo", "corro" y "correremos" → **"correr"**.

cantando -> cantar
canté -> cantar
correrá -> correr
pensado -> pensar
estrellado [verbo] -> estrellar
estrellado [adjetivo] -> estrellado
gatos -> gato
gata -> gato

Stemming

El stemming es un método simplificado de normalización de texto que corta directamente los sufijos de las palabras para reducirlas a una forma base, conocida como raíz (stem).

- A diferencia de la lematización, no utiliza análisis morfológicos complejos ni considera el contexto gramatical.

Estos algoritmos más simples principalmente consisten en el **corte directo de los sufijos de las palabras**.

cantando -> cant
canté -> cant
correrá -> corr
pensado -> pens
estrellado -> estrell
gatos -> gat

Utilización de stemming y lematización

Reducir el tamaño del vocabulario es clave para:

- **Visualización**: Facilita el análisis exploratorio de datos.
- **Generalización**: Mejora el rendimiento de los modelos al evitar sobreajustes.
- **Eficiencia**: Reduce los requisitos computacionales.

Ahora (modelos modernos de lenguaje):

- Los modelos usan **tokenización por subpalabras** lo que **reduce naturalmente el vocabulario** y **mitiga el problema OOV**.

La lematización o el stemming pueden aún ser útiles en:

- Tareas clásicas (búsqueda, conteo, clustering).
- Procesos interpretativos o explicativos.
- Modelos pequeños o específicos de dominio con datos limitados.

Errores Comunes en Preprocesamiento

1. **Preprocesar test diferente a train.**
2. **Eliminar stopwords sin considerar contexto**
    1. "No me gusta" → "gusta" (cambia el sentimiento)
3. **Tokenizar después de lowercase**
    1. "U.S.A." → "u.s.a." → ["u", "s", "a"]
4. **No manejar encoding correctamente**
    1. UTF-8 vs Latin-1 causa caracteres extraños
5. **Sobre-stemming sin validar**
    1. "university" → "univers" (pierde significado)

Exploración de texto

Exploración de Datos: Texto Libre vs. Datos Estructurados

En cualquier proyecto de **ciencia de datos**, la exploración inicial es crucial para entender el comportamiento y las características de los datos.

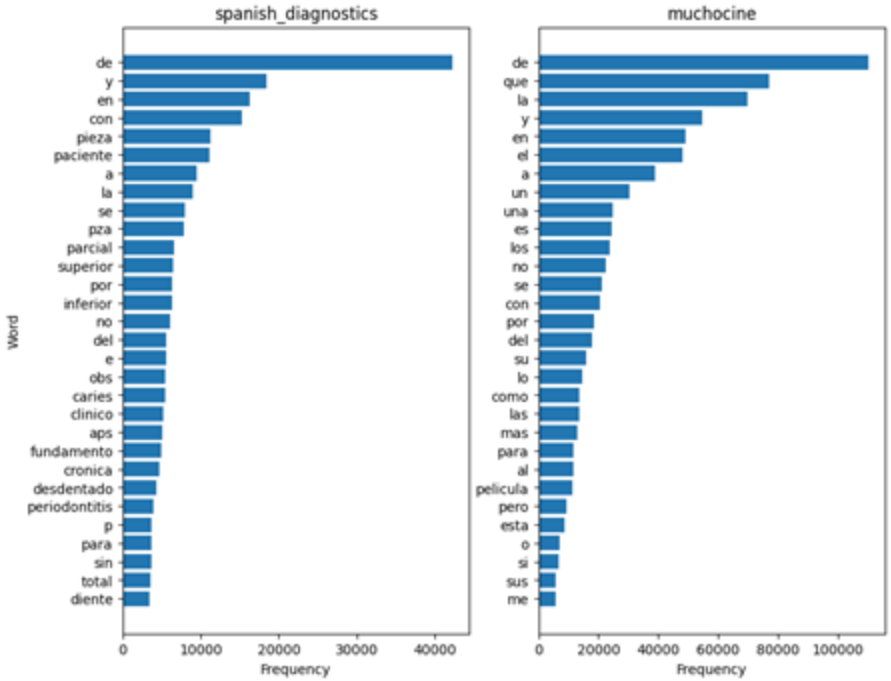
Diferencia clave:
Los datos de **texto libre no estructurado** requieren técnicas específicas, a diferencia de los datos **estructurados**.

La frecuencia de palabras

Después de tener calculado el vocabulario del corpus que se va a analizar se puede calcular la frecuencia de las palabras.

La frecuencia de palabras es la cantidad de veces que está la palabra dentro del corpus o el documento que se está analizando.

La distribución de la frecuencia de palabras nos puede ayudar a conocer cuál es el contenido y el contexto del corpus o de cada documento.



El tamaño del vocabulario

El tamaño del vocabulario es la cantidad de palabras distintas que presenta un corpus. Esto nos puede comunicar la complejidad que puede presentar el texto, pero se puede ver también confundida por el tamaño del corpus.

Table 1. Summary statistics of the corpora

	corpus		
Metric	German	English	Spanish
Articles count	59 539	22 372	12 058
Number of word tokens	20 437 502	12 093 145	51 337 854
Vocabulary size	497 256	144 550	374 877

Diversidad léxica

La diversidad léxica es un aspecto de la riqueza léxica y se refiere a la razón entre el tamaño del vocabulario y la cantidad total de tokens en el corpus.

Genre	Tokens	Types	Lexical diversity
skill and hobbies	82345	11935	0.145
humor	21695	5017	0.231
fiction: science	14470	3233	0.223
press: reportage	100554	14394	0.143
fiction: romance	70022	8452	0.121
religion	39399	6373	0.162

Tokens: The total number of words in a text.

Types: The number of unique words in a text.

Tokens: 6 (The, cat, sat, on, the, mat)
Types: 5 (The, cat, sat, on, mat)
TTR: 6/7

Concordancia

La concordancia muestra:

- Cada instancia de una palabra específica.

- Las palabras de contexto que la acompañan en el texto.

¿Cómo se utiliza?

- Ayuda a entender el **uso y significado** de una palabra en diferentes contextos.

- Facilita la identificación de **patrones lingüísticos** y **relaciones semánticas**.

```
>>> text1.concordance("monstrous")
Displaying 11 of 11 matches:
ong the former , one was of a most monstrous size . ... This came towards us ,
ON OF THE PSALMS . " Touching that monstrous bulk of the whale or ork we have r
ll over with a heathenish array of monstrous clubs and spears . Some were thick
d as you gazed , and wondered what monstrous cannibal and savage could ever hav
that has survived the flood ; most monstrous and most mountainous ! That Himmal
they might scout at Moby Dick as a monstrous fable , or still worse and more de
th of Radney .'" CHAPTER 55 Of the monstrous Pictures of Whales . I shall ere l
ing Scenes . In connexion with the monstrous pictures of whales , I am strongly
ere to enter upon those still more monstrous stories of them which are to be fo
ght have been rummaged out of this monstrous cabinet there is no telling . But
of Whale - Bones ; for Whales of a monstrous size are oftentimes cast up dead u
>>>
```

Bigramas y n-Gramas

Colocación: Secuencia de palabras que frecuentemente ocurren juntas con un significado específico.

Ejemplo: "hipertensión arterial" es una colocación común.

Los **bigrama** son pares de palabras que aparecen juntas con frecuencia.

Se pueden ampliar a **n-gramas** (pares de más de dos palabras) para capturar patrones de colocación más complejos.

Ejemplo: *"inteligencia artificial", "procesamiento de lenguaje natural"*

```
>>> text4.collocations()
United States; fellow citizens; four years; years
Government; General Government; American people; V
World; Almighty God; Fellow citizens; Chief Magist
God bless; every citizen; Indian tribes; public de
foreign nations; political parties
>>> text8.collocations()
would like; medium build; social drinker; quiet ni
long term; age open; Would like; easy going; finan
times; similar interests; Age open; weekends away;
presented; never married; single mum; permanent re
build
>>>
```

Nubes de palabras

Una **nube de palabras** es una visualización que mapea las palabras de un texto en un formato gráfico.

El **tamaño** de cada palabra refleja su **importancia relativa** en el texto, basada en métricas como:

- **Frecuencia bruta**: Cuántas veces aparece la palabra en el texto.

- **TF*IDF** (Term Frequency - Inverse Document Frequency): Medida que pondera la frecuencia de la palabra en el documento, considerando su frecuencia en otros documentos del corpus.

WordCloud de noticias de la categoría Negocios.