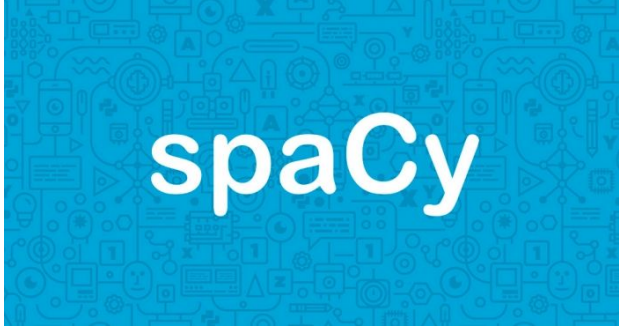
Procesamiento Natural del Lenguaje

Clase N° 4 - Reducción del vocabulario

Docente: James Tomalá Robles

¿Dónde estamos?

Librería Spacy



Diseñada específicamente para aplicaciones de producción y es conocida por su rapidez, eficiencia y facilidad de uso.

Spacy proporciona modelos preentrenados de diferentes idiomas

Algunos procesamientos:
Tokenización: Dividir texto en palabras, frases, párrafos, etc.
Stemming y lematización: Reducir palabras a sus raíces o formas base.
Etiquetado de partes del discurso (POS tagging)
Reconocimiento de entidades nombradas (NER)
Análisis sintáctico
Vectores de palabras (word vectors)

¿Cómo trabajar con SpaCy?

```
import spacy

# Cargar el modelo en español de spaCy
nlp = spacy.load('es_core_news_sm')
```

```
words = [token.text.lower() for token in doc
```

estandarización a minúsculas

```
if not token.is_stop
```

Eliminación de Stopwords

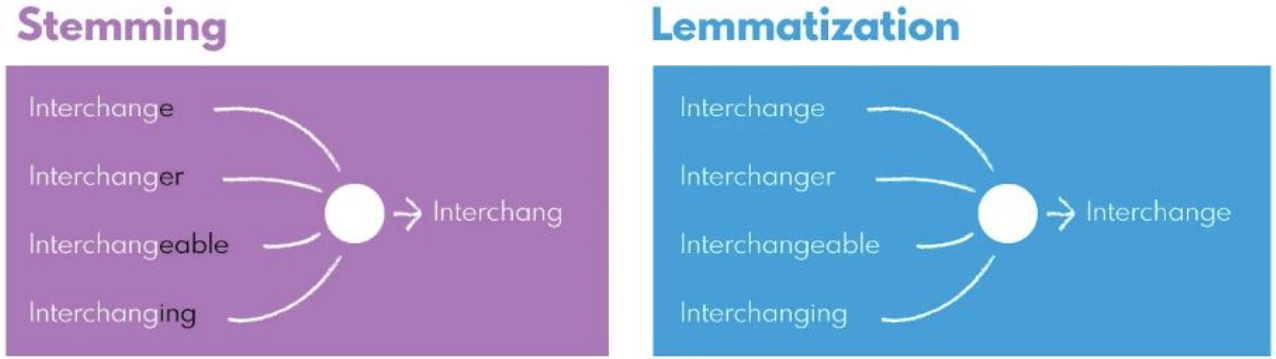
```
and not token.is_punct]
```

Limpieza de puntuaciones

Modelos SpaCy en español

es_core_news_sm:	Este es el modelo más pequeño y más rápido de spaCy para español. Es adecuado para tareas básicas de procesamiento de texto como tokenización, POS tagging (etiquetado gramatical), y análisis sintáctico.
es_core_news_md:	Este modelo es de tamaño medio y ofrece un equilibrio entre velocidad y precisión. Es ideal para aplicaciones que requieren un análisis más profundo del texto, como extracción de entidades nombradas y análisis de dependencias.
es_core_news_lg:	Es el modelo más grande y más preciso disponible para el español en spaCy. Es adecuado para tareas complejas de NLP que requieren un alto nivel de precisión, como la clasificación de texto, la generación de embeddings avanzados y el análisis semántico.

Reducción del vocabulario:

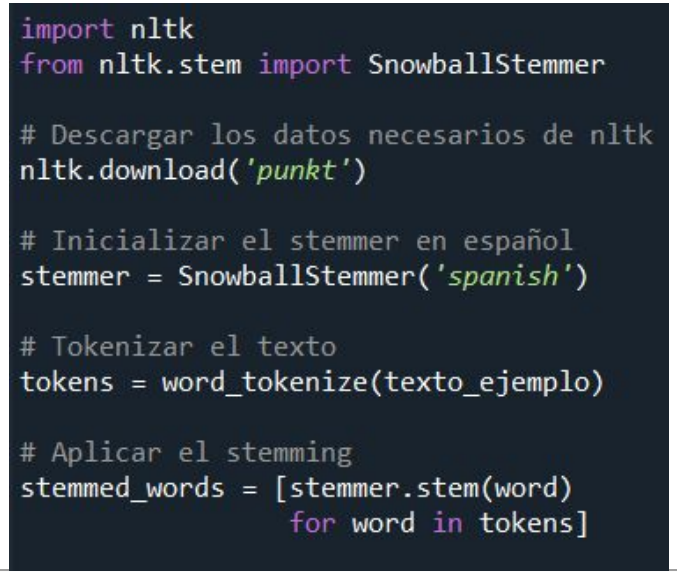


Stem (raíz): raíz de la palabra obtenida por radicalización.

Stemming vs Lemmatization

Lema: Forma base de la palabra

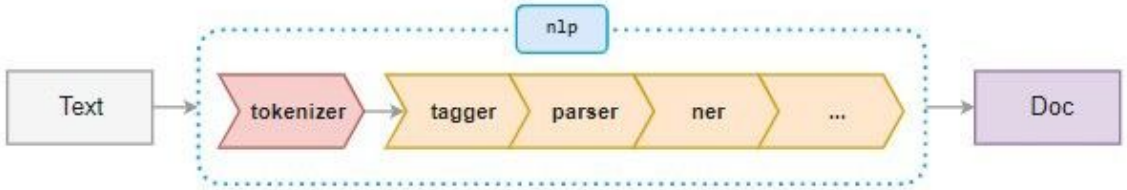
Algunos códigos para Stemming & Lematización

Stemming (nltk)	Lematización (SpaCy)
	

```
import nltk
from nltk.stem import SnowballStemmer

# Descargar los datos necesarios de nltk
nltk.download('punkt')

# Inicializar el stemmer en español
stemmer = SnowballStemmer('spanish')

# Tokenizar el texto
tokens = word_tokenize(texto_ejemplo)

# Aplicar el stemming
stemmed_words = [stemmer.stem(word)
                 for word in tokens]
```

```
[ token.lemma_ for token in doc]
```

la clase Doc de SpaCy



NAME	COMPONENT	CREATES	DESCRIPTION
tokenizer	`Tokenizer`	`Doc`	Segment text into tokens.
tagger	`Tagger`	`Doc[i].tag`	Assign part-of-speech tags.
parser	`DependencyParser`	`Doc[i].head`, `Doc[i].dep`, `Doc.sents`, `Doc.noun_chunks`	Assign dependency labels.
ner	`EntityRecognizer`	`Doc.ents`, `Doc[i].ent_iob`, `Doc[i].ent_type`	Detect and label named entities.
textcat	`TextCategorizer`	`Doc.cats`	Assign document labels.
...	custom components	`Doc._.xxx`, `Token._.xxx`, `Span._.xxx`	Assign custom attributes, methods or properties.

Doc es la clase central de SpaCy y representa un documento de texto procesado. Esta clase se utiliza para almacenar y manipular el texto después de haber sido procesado por el modelo de lenguaje de spaCy

IPP teclab onmex by SOCIAL LEARNING

Antes de ir al caso práctico:

Lematización es una técnica empleada en el PNL para lograr la raíz de cada token.

Verdadero ( )

Falso( )

Caso Práctico - Análisis de un libro

Extraer txt de un archivo en pdf .

Realizar limpieza, estandarización (minúsculas).

Realizar reducción por stemming(radicalización) y lematización.

Obtener una clasificación del texto por alguna librería.

Extrae texto de un txt

```python
### Extrae texto de un txt
def extract_text_from_txt(txt_path):
    """
    Extrae texto de un archivo TXT.

    :param txt_path: Ruta al archivo TXT.
    :return: Texto extraído del archivo TXT.
    """
    # Abre el archivo TXT en modo lectura
    with open(txt_path, 'r') as file:  # podría usar : , encoding='latin-1'
        # Lee todo el contenido del archivo
        extracted_text = file.read()

    return extracted_text

# Ejemplo de uso
txt_path = 'C:/data/cuento_astro_perdido.txt'
texto = extract_text_from_txt(txt_path)
print(texto)
```

Extrae texto de un PDF

```python
import PyPDF2

# Función para extraer texto del PDF usando PyPDF2
def extract_text_from_pdf(pdf_path):
    text = ""
    with open(pdf_path, 'rb') as file:
        reader = PyPDF2.PdfReader(file)
        for page_num in range(len(reader.pages)):
            page = reader.pages[page_num]
            text += page.extract_text()
    return text
```

Otras librerías más específicas

Transformers

Los **Transformers** son una arquitectura de modelos de aprendizaje profundo introducida en el artículo "Attention is All You Need" de Vaswani et al. en 2017. Esta arquitectura ha revolucionado el campo del procesamiento de lenguaje natural (NLP) y la inteligencia artificial en general debido a su capacidad para manejar secuencias de datos de manera eficiente y efectiva.

Encoders y Decoders:

Los encoders reciben la entrada y generan representaciones contextuales de cada palabra en la secuencia.

Los decoders usan estas representaciones para producir la salida secuencialmente.

Self-Attention:

Calcula la relevancia de cada palabra con respecto a las demás.

Esto permite capturar relaciones a largo plazo en la secuencia, algo que es difícil para los modelos basados en RNN (Redes Neuronales Recurrentes).

Cierre

https://quizizz.com/embed/quiz/667da5ff568780eb370b5e70