

ANEXOS: PROCESAMIENTO DE LENGUAJE NATURAL(NLP) CON SCIPY Y NLTK

SESION 2 / SEMANA 2

Anexo 01 : Tarea Aprendizaje 03

Alumno: [Juan Piero Vincha Loza]

1. ¿En que casos puede usarse el Procesamiento de Lenguaje Natural (NLP)? (2 PTOS)

El Procesamiento de Lenguaje Natural (NLP) puede usarse en casos como:

- **Análisis de sentimientos:** para identificar emociones o actitudes en textos como reseñas o comentarios en redes sociales.
- **Asistentes virtuales:** como chatbots o sistemas de respuesta automática que interactúan con los usuarios mediante lenguaje natural.
- **Traducción automática:** servicios como Google Translate que convierten textos de un idioma a otro.
- **Reconocimiento de voz:** convertir la voz en texto, como en asistentes personales tipo Alexa o Siri.

2. ¿Cuál es la diferencia entre SciPy y NLTK? (2 PTOS)

- **SciPy:** Es una librería en Python enfocada en operaciones matemáticas, científicas y de ingeniería, como álgebra lineal, optimización y procesamiento de señales.
- **NLTK:** Es una librería de Python para el Procesamiento de Lenguaje Natural (NLP) que proporciona herramientas para análisis de texto, tokenización, etiquetado gramatical y más.

SciPy es para matemáticas y ciencia en general, mientras que NLTK está especializado en la manipulación y análisis de lenguaje humano.

3. ¿En que aplicaciones se usa el Modelado de Lenguaje y tokenización de texto? (2 PTOS)

El modelado de lenguaje y la tokenización de texto se usan en:

- **Motores de búsqueda:** para entender las consultas de los usuarios y proporcionar resultados más relevantes.
 - **Generación de texto:** en aplicaciones como GPT que generan texto coherente a partir de un input.
 - **Corrección gramatical:** en herramientas que analizan el texto para sugerir correcciones o mejoras, como Grammarly.
 - **Chatbots y asistentes virtuales:** para interpretar el lenguaje y responder de forma apropiada.
4. ¿Cuál es la diferencia entre paquete MachineLearning y Deep Learning? (2 PTOS)

- **Machine Learning (ML):** es un subcampo de la inteligencia artificial que se enfoca en el desarrollo de algoritmos que pueden aprender de los datos y hacer predicciones o decisiones basadas en patrones detectados. Incluye técnicas como regresión lineal, árboles de decisión y SVM.
- **Deep Learning (DL):** es un subcampo de Machine Learning que utiliza redes neuronales profundas (con muchas capas) para analizar datos de manera más compleja, como en reconocimiento de imágenes o procesamiento de lenguaje natural. Las redes neuronales permiten a DL manejar grandes volúmenes de datos y realizar tareas más avanzadas.

Deep Learning es más complejo y requiere más datos y poder computacional que Machine Learning.

5. Implementar el siguiente programa en python que analiza los sentimientos de un texto “María está sentada en la playa, escuchando las olas del mar y el canto de las aves. Siente una profunda paz.” usando paquete NLTK y aplicando además técnicas de tokenización, etiquetado de partes, extracción de entidades. (12 PTOS). Rpta: Pantalla de Resultado. PD: no se olvide de descargar el paquete NLTK.

```
A juanito ~ /senati/machine-learning-main git-[P main]- >> cd A04
A juanito ~ /senati/machine-learning-main/A04 git-[P main]- >> python main.py
[nltk_data] Downloading package punkt to /home/juanito/nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data] /home/juanito/nltk_data...
[nltk_data] Package averaged_perceptron_tagger is already up-to-
[nltk_data] date!
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data] /home/juanito/nltk_data...
[nltk_data] Package maxent_ne_chunker is already up-to-date!
[nltk_data] Downloading package words to /home/juanito/nltk_data...
[nltk_data] Package words is already up-to-date!
[nltk_data] Downloading package vader_lexicon to
[nltk_data] /home/juanito/nltk_data...
[nltk_data] Package vader_lexicon is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] /home/juanito/nltk_data...
[nltk_data] Package stopwords is already up-to-date!
Tokens: ['María', 'está', 'sentada', 'en', 'la', 'playa', ',', 'escuchando', 'las', 'olas', 'del', 'mar', 'y', 'el', 'canto', 'de', 'las', 'aves', '.', 'Si
ente', 'una', 'profunda', 'paz', '.']

Filtered Tokens: ['María', 'sentada', 'playa', ',', 'escuchando', 'olas', 'mar', 'canto', 'aves', '.', 'profunda', 'paz', '.']

POS Tags: [('María', 'NNP'), ('está', 'NN'), ('playa', 'NN'), (',', ','), ('escuchando', 'JJ'), ('olas', 'NNS'), ('mar', 'VBP'), ('canto', 'JJ'), ('aves',
', 'NNS'), ('.', '.'), ('profunda', 'NN'), ('paz', 'NN'), ('.', '.')]

Named Entities:
GPE María

Sentiment Analysis: {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}

El sentimiento no es fuertemente positivo, lo que podría indicar una desalineación en el contexto de paz.
A juanito ~ /senati/machine-learning-main/A04 git-[P main]- >>
```

La imagen de la captura de pantalla esta en el archivo “A04.png” en el mismo repositorio, para una mejor visualización.

```
Welcome  tarea3.py x
tarea3.py > ...
1  import nltk
2  from nltk.tokenize import word_tokenize
3  from nltk import pos_tag, ne_chunk
4  from nltk.sentiment import SentimentIntensityAnalyzer
5  from nltk.corpus import stopwords
6
7  # Descargar recursos necesarios
8  nltk.download('punkt')
9  nltk.download('punkt_tab')
10 nltk.download('averaged_perceptron_tagger')
11 nltk.download('averaged_perceptron_tagger_eng')
12 nltk.download('maxent_ne_chunker')
13 nltk.download('maxent_ne_chunker_tab')
14 nltk.download('words')
15 nltk.download('vader_lexicon')
16 nltk.download('stopwords')
17
18 # Texto de ejemplo en español
19 text = "María está sentada en la playa, escuchando las olas del mar y el canto de las aves. Siente una profunda paz."
20
21 # Tokenización
22 tokens = word_tokenize(text, language='spanish')
23 print("Tokens:", tokens)
24
25 # Eliminación de stopwords
26 stop_words = set(stopwords.words('spanish'))
27 filtered_tokens = [word for word in tokens if word.lower() not in stop_words]
28 print("\nFiltered Tokens:", filtered_tokens)
29
30 # Etiquetado de Partes del Discurso (POS Tagging)
31 pos_tags = pos_tag(filtered_tokens)
32 print("\nPOS Tags:", pos_tags)
33
34 # Extracción de Entidades Nombradas
35 named_entities = ne_chunk(pos_tags)
36 print("\nNamed Entities:")
37 for entity in named_entities:
38     if hasattr(entity, 'label'):
39         print(entity.label(), ' '.join(c[0] for c in entity))
40
41 # Análisis de Sentimientos
42 sia = SentimentIntensityAnalyzer()
43 sentiment = sia.polarity_scores(text)
44 print("\nSentiment Analysis:", sentiment)
45
46 # Validación Semántica y Pragmática
47 if sentiment['compound'] > 0.5:
48     print("\nEl sentimiento de la oración es positivo, lo cual coincide con el contexto de paz y tranquilidad en la playa.")
49 else:
50     print("\nEl sentimiento no es fuertemente positivo, lo que podría indicar una desalineación con el contexto de paz.")
51
```