PUCE Virtual
EDUCACIÓN FLEXIBLE

Adquisición, gestión y gobernanza de datos

Tópicos de Textmining

Clase 4

MAESTRÍA EN
SISTEMAS DE INFORMACIÓN
Mención Data Science

La excelencia no se improvisa

INTRODUCCIÓN

El *text mining* o minería de texto se ha convertido en un área clave en el análisis de datos, debido a la creciente cantidad de información textual generada en el mundo digital. Este proceso implica la extracción de información útil y patrones significativos a partir de grandes volúmenes de datos no estructurados, como documentos, correos electrónicos y publicaciones en redes sociales. A través de técnicas de procesamiento del lenguaje natural (PLN), aprendizaje automático y análisis estadístico, el *text mining* permite a las organizaciones comprender mejor el contenido textual, facilitando la toma de decisiones informadas y la identificación de tendencias en diversos campos, desde la investigación académica hasta el marketing (Feldman & Sanger, 2007).

A medida que el volumen de datos textuales sigue aumentando, también lo hace la importancia de los temas relacionados con el *text mining*. Estos incluyen la clasificación de textos, la extracción de información, el análisis de sentimientos y la detección de temas. Cada uno de estos temas presenta sus propios desafíos y metodologías, lo que hace que el campo sea tanto diverso como dinámico. Por ejemplo, el análisis de sentimientos se utiliza ampliamente en el ámbito empresarial para evaluar la percepción de los consumidores sobre productos o servicios a través de sus comentarios en línea (Liu, 2012). En este contexto, el *text mining* no solo ofrece herramientas para analizar datos, sino que también proporciona un marco para abordar preguntas complejas y tomar decisiones estratégicas basadas en la evidencia textual.

RDA2

Esquematizar metodologías apropiadas para la obtención de modelos inductivos (predictivos) y deductivos (descriptivos) que faciliten los procesos de toma de decisiones basadas en datos en las instituciones.

Clase 4: Tópicos de Textmining

Recuperación de Información

La recuperación de información (RI) es una disciplina fundamental en el ámbito de la gestión del conocimiento y el acceso a datos, especialmente en un mundo donde la cantidad de información disponible en línea crece exponencialmente. En su esencia, la RI se refiere al proceso de buscar y recuperar información relevante a partir de un conjunto de datos. Este proceso implica el uso de algoritmos y técnicas que permiten a los usuarios localizar datos específicos en bases de datos, motores de búsqueda y sistemas de información. A medida que las organizaciones generan y almacenan grandes volúmenes de datos, la capacidad de realizar búsquedas efectivas y eficientes se convierte en una necesidad crítica (Manning et al., 2008).

Uno de los aspectos más interesantes de la recuperación de información es su aplicación en motores de búsqueda, que son la puerta de entrada a la vasta cantidad de contenido disponible en la web. Los motores de búsqueda utilizan diversas técnicas, como el análisis de texto, la indexación y la puntuación de relevancia, para ofrecer resultados que se alineen con las consultas de los usuarios. Esto no solo incluye texto, sino también imágenes, videos y otros tipos de datos. Además, la RI ha evolucionado para incorporar el aprendizaje automático y el procesamiento del lenguaje natural, lo que permite comprender mejor las intenciones detrás de las consultas de los usuarios y mejorar la precisión de los resultados (Croft et al., 2015). En este contexto, la recuperación de información no solo facilita el acceso a datos, sino que también empodera a los usuarios para tomar decisiones informadas en un entorno cada vez más complejo y saturado de información.

Sistemas de recuperación de información

Los sistemas de recuperación de información (SRI) son herramientas esenciales en la era digital, diseñadas para facilitar el acceso a grandes volúmenes de datos y permitir a los usuarios encontrar información relevante de manera eficiente. Estos sistemas funcionan mediante la organización, almacenamiento y recuperación de información, lo que les permite responder a consultas formuladas en lenguaje natural o mediante palabras clave. La funcionalidad de un SRI no solo depende de su capacidad para recuperar datos, sino también de la calidad de los resultados que ofrece, lo que implica una evaluación constante de su eficacia y eficiencia (Baeza-Yates & Ribeiro-Neto, 1999).

Un SRI típicamente consta de tres componentes principales: la **colección de datos**, el **índice** y el **módulo de recuperación**. La colección de datos es el conjunto de documentos y recursos que se gestionan, los cuales pueden incluir textos, imágenes, videos y más. El índice es una estructura de datos que permite un acceso rápido a la información, facilitando búsquedas más eficientes. Finalmente, el módulo de recuperación es el componente que

procesa las consultas del usuario y devuelve los resultados relevantes. Los SRI modernos incorporan técnicas avanzadas, como el procesamiento del lenguaje natural y el aprendizaje automático, para mejorar la interpretación de las consultas y la relevancia de los resultados (Manning et al., 2008). Esto ha permitido que los sistemas evolucionen, proporcionando respuestas más precisas y contextuales, lo que es particularmente útil en aplicaciones como motores de búsqueda, bases de datos académicas y sistemas de gestión de contenido.

Modelo Booleano en Recuperación de Información

El modelo booleano es uno de los enfoques más fundamentales en el campo de la recuperación de información y ha sido ampliamente utilizado desde sus inicios en la búsqueda de datos. Este modelo se basa en la lógica booleana, que utiliza operadores lógicos como "AND", "OR" y "NOT" para combinar términos de búsqueda y filtrar resultados. Su simplicidad y efectividad lo convierten en una herramienta poderosa para los usuarios que buscan información precisa en bases de datos y motores de búsqueda. Cuando un usuario realiza una consulta utilizando el modelo booleano, el sistema recupera documentos que cumplen con los criterios especificados por los operadores, lo que permite un mayor control sobre los resultados obtenidos (Salton & McGill, 1983).

Uno de los aspectos más destacados del modelo booleano es su capacidad para manejar consultas complejas. Por ejemplo, si un usuario busca información sobre "salud" y "nutrición", puede utilizar la expresión "salud AND nutrición" para recuperar solo los documentos que contengan ambos términos. De igual manera, si desea excluir un término, puede emplear "salud NOT diabetes" para filtrar documentos relacionados con la diabetes. Sin embargo, a pesar de su utilidad, el modelo booleano también presenta limitaciones, como la dificultad para manejar consultas más matizadas y la falta de puntuación de relevancia en los resultados. Esto significa que un documento que contenga los términos de búsqueda podría ser devuelto incluso si no es particularmente relevante para la consulta (Baeza-Yates & Ribeiro-Neto, 1999). A pesar de estas limitaciones, el modelo booleano sigue siendo una base esencial para entender otros enfoques más avanzados en la recuperación de información.

Métricas para medir la efectividad de un modelo de recuperación de información: Precisión y Recall

La evaluación de la efectividad de un modelo de recuperación de información es crucial para determinar su rendimiento en la satisfacción de las consultas de los usuarios. Dos de las métricas más importantes para esta evaluación son la **precisión** y el **recall**. Ambas métricas ofrecen una visión clara sobre cómo un modelo maneja la información relevante y, al mismo tiempo, permiten identificar áreas de mejora.

La **precisión** se define como la proporción de documentos relevantes recuperados frente al total de documentos que el modelo ha devuelto. Su fórmula se expresa de la siguiente manera:

$$\boldsymbol{precisión} = \frac{\boldsymbol{Verdaderos\ Positivos\ (VP)}}{\boldsymbol{Verdaderos\ Positivos\ (VP) + Falsos\ Positivos\ (FP)}}$$

Donde:

- **Verdaderos Positivos (VP)** son los documentos relevantes que el modelo ha recuperado.
- **Falsos Positivos (FP)** son los documentos que el modelo ha recuperado pero que no son relevantes.

Una alta precisión indica que el modelo es eficiente al seleccionar información relevante y minimiza la cantidad de datos irrelevantes que se presentan al usuario. Esto es especialmente importante en contextos donde se necesita información altamente específica, como en aplicaciones legales o médicas (Manning et al., 2008).

Por otro lado, el **recall** mide la capacidad del modelo para recuperar todos los documentos relevantes en la colección. La fórmula para el recall es:

$$\boldsymbol{recall} = \frac{\boldsymbol{Verdaderos\ Positivos\ (VP)}}{\boldsymbol{Verdaderos\ Positivos\ (VP) + Falsos\ Negativos\ (FP)}}$$

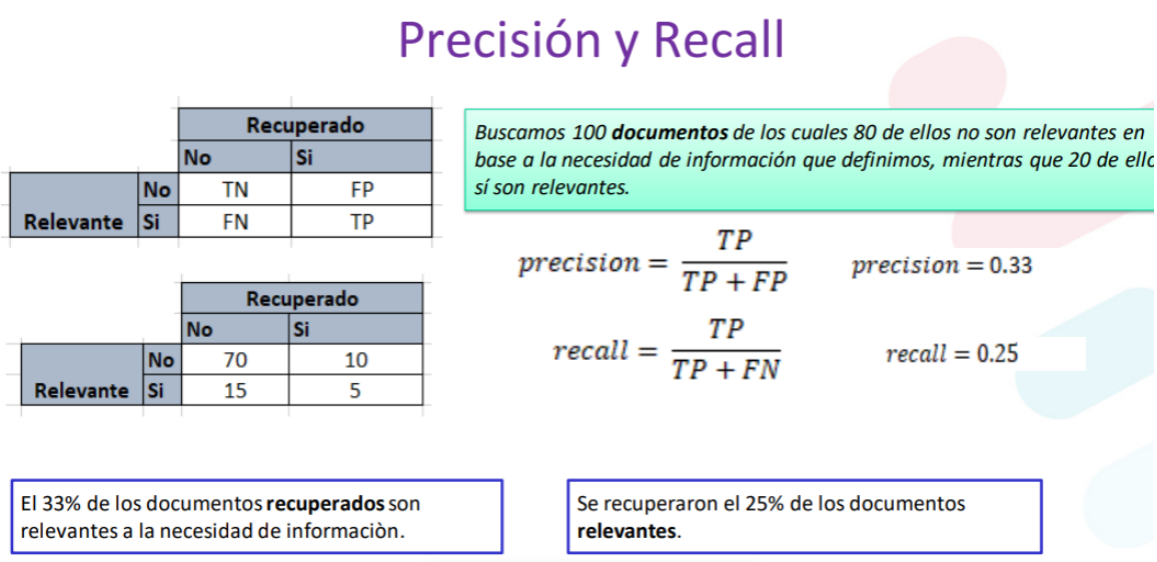
Donde:

- **Falsos Negativos (FN)** son los documentos relevantes que el modelo no ha logrado recuperar.

A continuación, se muestra un ejemplo del uso de las métricas de precisión y recall:

Figura 1

Precisión y Recall en Sistemas de Recuperación de Información



Fuente: Elaboración propia

Un alto recall significa que el sistema puede identificar la mayoría de los documentos relevantes, aunque esto podría implicar un aumento en el número de falsos positivos. Esta métrica es particularmente valiosa en situaciones donde es crítico no omitir información relevante, como en investigaciones científicas o en la búsqueda de literatura médica (Salton & McGill, 1983).

Es importante señalar que precisión y recall son métricas que a menudo se contraponen; aumentar una puede resultar en la disminución de la otra. Por esta razón, se utilizan métricas combinadas como la F1-score, que proporciona un equilibrio entre ambas y se calcula de la siguiente manera:

$$f1 - score = \frac{Precision * Recall}{Precision + Recall}$$

Esta métrica es útil en escenarios donde se requiere un equilibrio entre la precisión y el recall, permitiendo a los analistas evaluar de manera integral el rendimiento del modelo (Baeza-Yates & Ribeiro-Neto, 1999).

Índice Invertido: Estructura y Funcionamiento

El índice invertido es una de las estructuras de datos más cruciales en la recuperación de información, especialmente en motores de búsqueda y sistemas de gestión de datos textuales. Su diseño permite una búsqueda rápida y eficiente de documentos al asociar cada

término con los documentos en los que aparece. Esta técnica se utiliza ampliamente en aplicaciones como Google, bibliotecas digitales y sistemas de gestión de contenido, facilitando la recuperación de información relevante de grandes volúmenes de datos.

Construcción del Índice Invertido

Para ilustrar cómo funciona un índice invertido, consideremos un ejemplo sencillo con una colección de documentos:

- **Documento 1**: "La inteligencia artificial mejora la recuperación de información."
- **Documento 2**: "La inteligencia artificial puede transformar el sector de la salud."
- **Documento 3**: "La recuperación de información es esencial para la toma de decisiones."

Primero, necesitamos extraer los términos únicos de estos documentos. Los términos en este caso serían: "inteligencia", "artificial", "mejora", "recuperación", "de", "información", "puede", "transformar", "sector", "salud", "esencial", "para", "la", "toma", "decisiones".

A continuación, construimos el índice invertido. Para ello, se cuenta la cantidad de documentos en los que aparece cada término. La representación del índice invertido se puede mostrar de la siguiente manera:

Tabla 1

Matriz Incidencia Termino Documento y Diccionario de Términos

Término	Documentos
inteligencia	1, 2
artificial	1, 2
mejora	1
recuperación	1, 3
de	1, 2, 3
información	1, 3
puede	2
transformar	2
sector	2
salud	2
esencial	3
para	3
la	1, 2, 3
toma	3
decisiones	3

Fuente: Elaboración propia

Este índice permite que, al realizar una consulta, el sistema identifique rápidamente qué documentos contienen los términos buscados.

Ejemplo de búsqueda

Supongamos que un usuario busca "inteligencia y recuperación". El sistema consultaría el índice invertido y encontraría:

- **inteligencia**: Documentos 1 y 2
- **recuperación**: Documentos 1 y 3

La intersección de estos conjuntos de documentos (Documentos 1 y 3) indicaría que son los resultados relevantes para la consulta del usuario. Así, el índice invertido proporciona una forma eficiente de manejar búsquedas complejas.

Ventajas del Índice Invertido

Una de las principales ventajas del índice invertido es su capacidad para escalar. A medida que se agregan más documentos, el índice puede actualizarse de manera eficiente sin necesidad de recorrer toda la colección. Además, la estructura permite realizar búsquedas rápidas, ya que se puede acceder directamente a la lista de documentos para cada término, en lugar de tener que examinar cada documento individualmente.

En resumen, el índice invertido es una herramienta esencial en la recuperación de información moderna. Su capacidad para organizar y acceder rápidamente a grandes cantidades de datos textuales lo convierte en la base sobre la cual se construyen muchas aplicaciones de búsqueda actuales (Manning et al., 2008).

Análisis de Sentimientos: Conceptos y Ejemplos

El análisis de sentimientos es una técnica de procesamiento de lenguaje natural (NLP, por sus siglas en inglés) que permite identificar y extraer información subjetiva de un texto. Esta técnica se utiliza comúnmente para determinar la actitud del autor hacia un tema específico, ya sea positiva, negativa o neutral. El análisis de sentimientos se aplica en diversos campos, incluyendo el marketing, la atención al cliente, la investigación de mercados y las redes sociales.

¿Cómo Funciona el Análisis de Sentimientos?

El proceso de análisis de sentimientos generalmente incluye varios pasos:

- **Recolección de datos**: Se obtienen textos de diversas fuentes, como reseñas de productos, publicaciones en redes sociales o encuestas.

- **Preprocesamiento**: Se normaliza el texto, eliminando ruido como signos de puntuación, stopwords y aplicando técnicas de stemming o lematización.
- **Clasificación de sentimientos**: Se utilizan algoritmos de *machine learning* o léxicos de sentimientos para clasificar los textos. Los léxicos son listas de palabras con etiquetas de sentimientos predefinidas.
- **Evaluación**: Se mide la efectividad del modelo mediante métricas como precisión, recall y F1-score.

Ejemplo Práctico de Análisis de Sentimientos

Imaginemos que queremos analizar los sentimientos expresados en tweets sobre un nuevo producto. A continuación, se presenta una pequeña colección de tweets:

- "Me encanta este nuevo teléfono. ¡Es increíble!"
- "No me gusta el diseño de este teléfono. Muy feo."
- "Es un teléfono decente, pero podría ser mejor."
- "¡El servicio al cliente fue excelente!"

Podemos clasificar estos tweets como sigue:

- **Tweet 1**: Positivo
- **Tweet 2**: Negativo
- **Tweet 3**: Neutral
- **Tweet 4**: Positivo

Uso de Lógica de Sentimientos

Para realizar un análisis de sentimientos más formal, podemos utilizar un enfoque basado en un léxico de sentimientos, como VADER (Valence Aware Dictionary and sEntiment Reasoner), que está diseñado para el análisis de sentimientos en textos cortos

Cálculo de un Puntaje de Sentimiento:

El puntaje de sentimiento se puede calcular utilizando una simple fórmula:

$$Puntaje = \frac{\#\ palabras\ positivas - \#\ palabras\ negativas}{\#\ total\ palabras}$$

Aplicando esta fórmula a los tweets de ejemplo:

1. **Tweet 1**:
   - Palabras positivas: "encanta", "increíble"
   - Palabras negativas: 0

    - Puntaje: $(2-0)/6 \approx 0.33$
2. **Tweet 2**:
    - Palabras positivas: 0
    - Palabras negativas: "gusta", "feo"
    - Puntaje: $(0-2)/6 \approx -0.33$
3. **Tweet 3**:
    - Palabras positivas: 0
    - Palabras negativas: 0
    - Puntaje: $(0-0)/7=0$
4. **Tweet 4**:
    - Palabras positivas: "excelente"
    - Palabras negativas: 0
    - Puntaje: $(1-0)/5=0.2$

Interpretación de Resultados

- Un puntaje positivo indica un sentimiento positivo, mientras que un puntaje negativo indica un sentimiento negativo. Un puntaje cercano a cero sugiere un sentimiento neutral.
- En nuestro ejemplo, los tweets 1 y 4 tienen sentimientos positivos, el tweet 2 tiene un sentimiento negativo y el tweet 3 tiene un sentimiento neutral.

Normalización de Texto

La **normalización de texto** es un proceso fundamental en el campo del procesamiento de lenguaje natural (NLP) y la recuperación de información. Su objetivo es transformar el texto en un formato estandarizado que facilite su análisis y comparación. Este proceso es crucial para mejorar la precisión de los sistemas de búsqueda, la minería de datos y el análisis de sentimientos, entre otros. A continuación, se presenta un video explicativo sobre la normalización del texto del profesor Ghassemi: Normalización de texto.

¿Por qué es necesario normalizar el texto?

El texto natural puede presentar variaciones que dificultan su interpretación. Estas variaciones incluyen diferencias en mayúsculas y minúsculas, acentos, espacios en blanco y puntuación. Normalizar el texto ayuda a reducir la complejidad y a obtener resultados más precisos en el análisis. Por ejemplo, las palabras "gato", "Gato" y "GATO" deben ser tratadas como equivalentes en un sistema que busca contar la frecuencia de esa palabra.

Pasos Comunes en la Normalización de Texto

- **Conversión a minúsculas**: Cambiar todas las letras a minúsculas para asegurar la uniformidad. Por ejemplo, el texto "El Gato Come" se convierte en "el gato come".

$$texto\ normalizado = tolower(texto\ original)$$

- **Eliminación de Puntuación**: Retirar signos de puntuación que no aportan significado a la búsqueda o análisis. Por ejemplo, "Hola, mundo!" se convierte en "Hola mundo".

$$texto\ normalizado = regex\_replace(texto\ original, patro´n\ de\ puntuacio´n, "")$$

- **Eliminación de Espacios en Blanco**: Quitar espacios adicionales y asegurar que solo haya un espacio entre palabras. Por ejemplo, " Hola mundo " se convierte en "Hola mundo".

$$texto\ normalizado = trim(regex\_replace(texto\ original, "s+", ""))$$

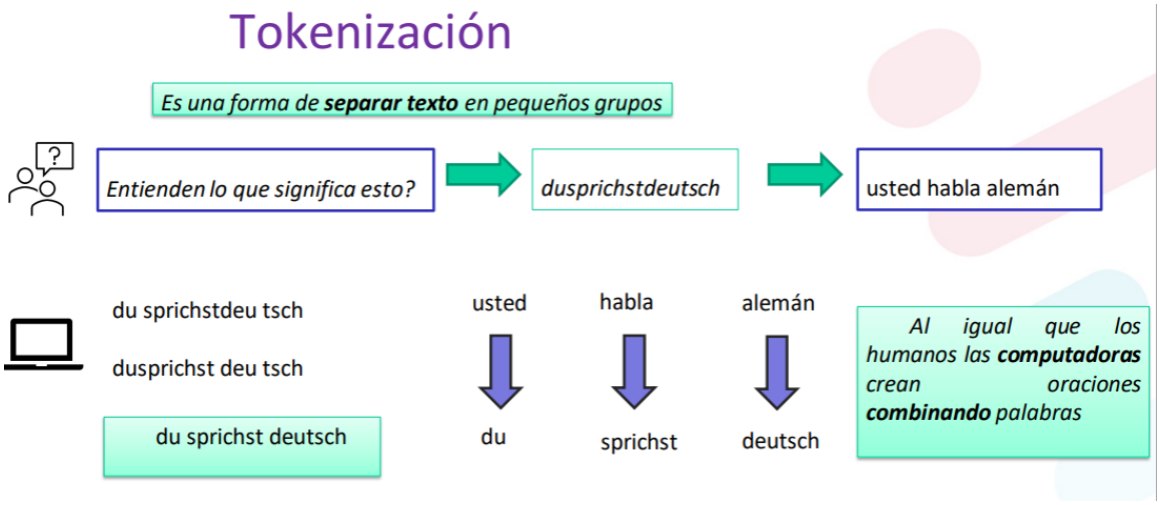
- **Stemming o Lematización**: Reducir las palabras a su raíz o forma básica. Por ejemplo, "correr", "corriendo" y "corre" pueden normalizarse a "corr".

$$palabra\ normalizada = stem(palabra\ original)$$

A continuación, se explica la necesidad de tokenizar mediante una infografía:

Figura 2

Proceso de tokenización.



Fuente: Elaboración propia

Ejemplo Práctico

Consideremos un conjunto de oraciones que se desea normalizar:

- "El Gato come."
- "gato, gato y Gato."

- "Correr, Corriendo y Corre."

Después de aplicar los pasos de normalización, el resultado sería:

- **Original**: "El Gato come."
- **Normalizado**: "el gato come"
- **Original**: "gato, gato y Gato."
- **Normalizado**: "gato gato y gato"
- **Original**: "Correr, Corriendo y Corre."
- **Normalizado**: "corr corr y corr"

Este proceso permite que un sistema de análisis trate estas entradas de manera más eficiente, lo que da como resultado un conteo correcto de las palabras relevantes.

Importancia de la Normalización

La normalización de texto es crucial para mejorar la calidad de los datos en análisis posteriores. Al estandarizar el texto, se reduce la variabilidad que podría interferir con los algoritmos de búsqueda y análisis, lo que permite obtener resultados más confiables y significativos. Sin una normalización adecuada, los sistemas podrían interpretar erróneamente la información, lo que llevaría a conclusiones incorrectas.

En conclusión, la normalización de texto es un proceso esencial que facilita la comprensión y el análisis del lenguaje natural. Al aplicar técnicas de normalización, se asegura que el texto sea uniforme y consistente, lo que mejora significativamente el rendimiento de las aplicaciones de procesamiento de datos (Manning et al., 2008).

Los **Large Language Models** (LLMs) son modelos de *machine learning* preentrenados con una gran cantidad de información y son altamente efectivos en tareas de generación de texto, traducción, generación de código, análisis de sentimientos, entre otras. En el siguiente video, puede ver alguna aplicación práctica de estos modelos, desde el minuto 30 hasta el 34: Ejemplos de Large Language Models.

A continuación, se muestra un ejemplo completo desarrollado en Python que realiza todo el proceso de análisis exploratorio y normalización de texto sobre un conjunto de datos de tweets. Primero, asegúrate de tener instaladas las bibliotecas necesarias. Si no las tienes, puedes instalarlas usando pip:

```
! pip install pandas nltk matplotlib seaborn
```

Figura 3

Código en Python:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from nltk.corpus import stopwords
import re

# Descargar stopwords de NLTK
nltk.download('stopwords')
stop_words = set(stopwords.words('spanish'))

# Cargar el conjunto de datos de tweets
# Para este ejemplo, supongamos que tienes un archivo CSV
llamado "tweets.csv"
# con una columna llamada 'tweet'.
# df = pd.read_csv('tweets.csv')  # Descomenta esta línea si
tienes el CSV
# Para fines de este ejemplo, crearemos un DataFrame de
ejemplo:

data = {
    'tweet': [
        "¡Me encanta Python! #programación #python",
        "La programación es divertida :)",
        "No puedo creer lo fácil que es aprender a programar!
:)",
        "Hoy es un gran día para programar.",
        "¿Quién más ama programar en Python?",
        "La vida es corta, ¡aprende a programar!",
        "No más errores en mi código, ¡por fin! #python
#programación"
    ]
}

df = pd.DataFrame(data)

# 1. Análisis exploratorio
print("Primeras filas del DataFrame:")
print(df.head())
```

```
# Longitud de los tweets
df['length'] = df['tweet'].apply(len)

# Visualización de la longitud de los tweets
plt.figure(figsize=(10, 6))
sns.histplot(df['length'], bins=10, kde=True)
plt.title('Distribución de la longitud de los tweets')
plt.xlabel('Longitud del tweet')
plt.ylabel('Frecuencia')
plt.show()

# 2. Normalización de texto
def normalize_text(text):
    # Conversión a minúsculas
    text = text.lower()
    # Eliminación de URLs
    text = re.sub(r'http\S+|www\S+|https\S+', '', text,
flags=re.MULTILINE)
    # Eliminación de menciones
    text = re.sub(r'@\w+', '', text)
    # Eliminación de hashtags
    text = re.sub(r'#\w+', '', text)
    # Eliminación de puntuación
    text = re.sub(r'[^\w\s]', '', text)
    # Eliminación de números
    text = re.sub(r'\d+', '', text)
    # Eliminación de stopwords
    text = ' '.join(word for word in text.split() if word not
in stop_words)
    return text

# Aplicar normalización
df['normalized_tweet'] = df['tweet'].apply(normalize_text)

# Mostrar tweets normalizados
print("\nTweets normalizados:")
print(df[['tweet', 'normalized_tweet']])

# 3. Análisis de la frecuencia de palabras
all_words = ' '.join(df['normalized_tweet']).split()
word_freq = pd.Series(all_words).value_counts()
```

```
# Visualización de la frecuencia de palabras
plt.figure(figsize=(10, 6))
word_freq.head(10).plot(kind='bar')
plt.title('Frecuencia de las palabras más comunes')
plt.xlabel('Palabras')
plt.ylabel('Frecuencia')
plt.xticks(rotation=45)
plt.show()
```

Explicación del Código

1. **Importación de bibliotecas**: Se importan las bibliotecas necesarias para el análisis de datos y el procesamiento de texto.
2. **Carga de datos**: Se crea un DataFrame de ejemplo con tweets. En un caso real, se puede cargar un archivo CSV que contenga los tweets.
3. **Análisis exploratorio**:
    - Se imprime la cabeza del DataFrame.
    - Se calcula la longitud de cada tweet y se visualiza la distribución de longitudes con un histograma.
4. **Normalización de texto**:
    - Se define la función `normalize_text`, que convierte el texto a minúsculas, elimina URLs, menciones, hashtags, puntuación, números y stopwords en español.
    - Se aplica esta función a la columna de tweets originales y se almacena el resultado en una nueva columna.
5. **Análisis de frecuencia de palabras**:
    - Se concatenan todos los tweets normalizados en una sola cadena, se divide en palabras y se cuentan las frecuencias.
    - Se visualiza la frecuencia de las palabras más comunes en un gráfico de barras.

Resultados

Este script proporciona una introducción al análisis exploratorio y la normalización de texto en un conjunto de datos de *tweets*. Puedes adaptarlo y ampliarlo según tus necesidades específicas, como agregar más pasos de análisis o modificar las técnicas de normalización.

Referencias

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Addison-Wesley.

Croft, W. B., Metzler, D., & Strohman, T. (2015). *Search engines: Information retrieval in practice*. Addison-Wesley.

Feldman, R., & Sanger, J. (2007). *The text mining handbook: Advanced approaches in analyzing unstructured data*. Cambridge University Press.

Liu, B. (2012). *Sentiment analysis and opinion mining*. Morgan & Claypool.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. MIT Press.

Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. McGraw-Hill.

PUCE Virtual
EDUCACIÓN FLEXIBLE

La excelencia no se improvisa

síguenos

MDASC - 03 - 251