Web Scraping en Wikipedia

Práctica Python
Lenguajes de Programación para Análisis de Datos (curso 2024-2025)

Objetivo de la Práctica:

El objetivo de esta práctica es que los alumnos desarrollen habilidades en técnicas de web scraping y análisis comparativo de datos a través de la extracción de información de Wikipedia. Los estudiantes aprenderán a automatizar la recopilación de datos de categorías de artículos, explorar la jerarquía de categorías y subcategorías, y comparar contenidos entre versiones en diferentes idiomas (español e inglés). Esta práctica les permitirá adquirir competencias en:

- Extracción de información de páginas web.
- Procesamiento y análisis de datos multilingües.
- Creación de estructuras jerárquicas (árboles de categorías) y visualización de datos.
- Buenas prácticas de scraping.

Fases de la Práctica:

1. Selección de una categoría inicial en Wikipedia: Elegir una categoría de artículos en Wikipedia en español para explorar. Esta categoría servirá como punto de partida para el resto de la práctica.

2. Extracción de la lista de artículos y subcategorías: Realizar scraping para obtener la lista de artículos y subcategorías de la categoría seleccionada. Esto incluirá extraer títulos de artículos y nombres de subcategorías desde la página de Wikipedia.

3. Construcción de la jerarquía de categorías: Crear un árbol o estructura jerárquica que muestre la relación entre la categoría principal, sus subcategorías y los artículos. Recorrer las subcategorías de manera recursiva.

4. Conteo de artículos en cada subcategoría: Contar los artículos de cada subcategoría y mostrar el resultado en un formato claro y ordenado, fácil de entender. Esta tarea ayudará a procesar y organizar datos obtenidos mediante scraping.

5. Comparación con la misma categoría en inglés: Buscar la misma categoría en Wikipedia en inglés y repetir las fases anteriores. Comparar el número de artículos y subcategorías entre ambos idiomas.

6. Identificación de artículos coincidentes y únicos: Comparar las listas de artículos en ambos idiomas para identificar cuáles están presentes en ambas versiones y cuáles son exclusivos de uno u otro idioma. Esta fase implica procesamiento de texto para buscar coincidencias.

7. Análisis comparativo de artículos coincidentes: Para los artículos que existen en ambas versiones (español e inglés), comparar las secciones del índice de cada artículo y analizar las diferencias en términos de contenido (cantidad de palabras por sección, presencia de secciones adicionales, etc.).

8. Visualización y presentación de resultados: Organizar y visualizar los resultados obtenidos, ya sea mediante gráficos, tablas o diagramas de árbol. La presentación debe destacar las diferencias más notables y proporcionar conclusiones basadas en el análisis.

Scraping ético. Transparencia respecto al objetivo del bot:

Para que los administradores del sitio web identifiquen las peticiones de scraping como legítimas, y evitar problemas o que puedan implementarse bloqueos (siempre que las peticiones sean razonables) se recomienda que cada petición de scraping especifique un valor de "**`User-Agent`**" que identifique al alumno, así como el propósito educativo del bot de scraping, también se puede indicar un correo electrónico de contacto para que los administradores pueden comunicarse con el propietario del bot en caso de que lo consideren oportuno. Un ejemplo de petición con "**`request`**" que incluya la información de "**`User-Agent`**" podría ser el siguiente:

```
import request

headers = {
    'User-Agent': 'WikipediaEduBot/1.0 (User:UseID; mailto:nomUser@email.com)'
}

ur='dirección de la página para scrapear'
response = requests.get(url, headers=headers)
```

Para mayor transparencia, cada alumno puede crear una cuenta de usuario en Wikipedia e indicar en el '**`header`**' su ID de usuario y el email con el que se ha registrado, no tiene porqué ser el correo personal habitual, se puede crear uno específico o usar un email secundario para este propósito.

El objetivo de estas medidas es ser respetuosos y transparentes al hacer scraping. Especificar el propósito educativo y un medio de contacto en el "**`User-Agent`**" es una buena práctica para realizar scraping de forma responsable y transparente. Registrar una cuenta de Wikipedia y usar el nombre de usuario en el "**`User-Agent`**" es opcional, pero puede ayudar a demostrar que las intenciones son claras y educativas.

También se deberá leer y respetar los ficheros "**robots.txt**", en ambas versiones, inglés y español, de la wikipedia, dichos archivos se pueden encontrar en la direcciones:

- https://en.wikipedia.org/robots.txt
- https://es.wikipedia.org/robots.txt

Notar que ambos documentos "**robots.txt**" disponen de un amplio listado de URLs con la etiqueta "`Disallow:`", se debe evitar acceder mediante scraping a dichas páginas.

Requisitos de entrega:

Cada alumno deberá presentar los siguientes elementos al finalizar la práctica:

1. Scripts de Python: Los alumnos deben entregar un script de Python separado para cada una de las tareas principales de la práctica, asegurándose de que cada script esté bien documentado y sea funcional. Los scripts deben estar nombrados de forma clara para indicar qué tarea realizan (por ejemplo, extraccion_categorias.py, comparacion_articulos.py, etc.). Es importante que cada script incluya comentarios que expliquen el propósito de las funciones principales, así como cualquier paso importante en el proceso de extracción, procesamiento o análisis de datos.

2. Memoria en formato PDF: Se deberá entregar una memoria bien estructurada en formato PDF, que incluya:
   - Introducción: Breve descripción del objetivo de la práctica y el enfoque adoptado.
   - Metodología: Explicación de los pasos seguidos para llevar a cabo cada tarea, destacando las técnicas de web scraping, procesamiento de datos y comparación entre idiomas. Esta sección debe incluir detalles técnicos sobre el uso de las librerías, las dificultades encontradas y cómo se resolvieron.
   - Resultados: Presentación de los resultados obtenidos, con gráficas y tablas que ayuden a visualizar y comparar los datos de forma clara. Los gráficos deben estar correctamente etiquetados y acompañados de una breve explicación que los contextualice.
   - Conclusiones: Análisis de los resultados y las diferencias observadas entre las versiones en diferentes idiomas de Wikipedia. Se pueden incluir observaciones adicionales, como patrones encontrados o hipótesis sobre las razones de las diferencias.
   - Anexos (opcional): Cualquier información adicional que consideren relevante, como fragmentos de código explicativos, tablas completas de datos, etc.

3. Presentación para exponer el trabajo en clase: Los alumnos deberán preparar una presentación en formato digital (p.e.:, PowerPoint o similar) que resuma los resultados de su estudio para exponerlos en clase. La presentación debe ser clara, concisa y no exceder los 10 minutos de exposición por alumno.

La presentación debe incluir:

- Introducción breve al proyecto y su objetivo.
- Metodología resumida, destacando los puntos clave y técnicas utilizadas.
- Resultados visuales como gráficos y tablas que ilustren claramente los hallazgos más importantes.
- Conclusiones destacadas y, si es posible, preguntas para el debate o reflexiones adicionales sobre el análisis comparativo entre las versiones en español e inglés de Wikipedia.

Recomendaciones:

Otras consideraciones sobre el acceso automatizado a la Wikipedia:

- Limitar la frecuencia de las solicitudes: usar la librería "**`time`**" y el método "**`time.sleep(seg)`**" para no saturar el servidor. Desde el campus, se recomienda usar la conexión de datos del móvil para que los accesos sean desde diferentes IPs.

- Evitar patrones sospechosos: Acceder de forma frecuente y repetida a la misma página puede interpretarse como un ataque, se recomienda planificar y distribuir las peticiones. En caso de tener que repetir una petición se recomienda hacerla de forma aleatoria y con pausas variables para no crear un patrón fijo. Se puede usar "**`random.uniform(mim,max)`**" para modificar los tiempos de espera.

- Manejar errores de conexión y limitar reintentos: incluir mecanismos para manejar errores de conexión, como "**`try...except`**", y limitar los reintentos si una solicitud falla. Esto evita que el script se quede en un bucle de reintentos que pueda saturar el servidor.

```
1 try:
2     response = requests.get(url, headers=headers)
3     response.raise_for_status()  # Verificar si hubo algún error en la solicitud
4 except requests.exceptions.HTTPError as err:
5     print(f"Error HTTP: {err}")
6     # gestionar error
7 except requests.exceptions.RequestException as e:
8     print(f"Error de conexión: {e}")
9     # gestionar error
```

- Considerar el uso de un tiempo de espera (**`timeout`**): Al hacer scraping, es importante no dejar que las solicitudes se queden colgadas si el servidor no responde. Añadir un "**`timeout`**" permite que la solicitud falle rápidamente si no se obtiene una respuesta en un tiempo razonable (por ejemplo, 10 segundos).

```
12 response = requests.get(url, headers=headers, timeout=10)
```

- Descartar dalton innecesarios: Procurar no descargar imágenes, datos multimedia u otros similares para ahorrar ancho de banda y no sobrecargar el servidor.

Familiarizarse con la estructura HTML de las páginas de Wikipedia:

Antes de comenzar con la extracción de datos, se recomienda que los alumnos se familiaricen con la estructura HTML de las páginas de Wikipedia que van a analizar. Esto les ayudará a identificar fácilmente los elementos que necesitan extraer, tales como:

1. Enlaces a artículos: En las páginas de categorías, los artículos que pertenecen a esa categoría aparecen como enlaces. Hay que saber como localizar estos enlaces en el HTML (generalmente son elementos <a> dentro de listas <li> o específicos) y extraer la URL y el título del artículo.

2. Enlaces a subcategorías: Además de los artículos, las páginas de categorías de Wikipedia suelen tener subcategorías que están enlazadas de manera jerárquica. Es importante aprender a distinguir entre enlaces a artículos y enlaces a subcategorías, para poder construir la jerarquía de categorías de forma correcta.

3. Enlaces interlinguales: En la barra lateral de las páginas de Wikipedia, suelen aparecer enlaces a versiones del mismo artículo en diferentes idiomas. Estos enlaces interlinguales permiten identificar el artículo equivalente en otro idioma (por ejemplo, español e inglés). Los alumnos deberán aprender a localizar estos enlaces para comparar artículos en diferentes idiomas.

Para familiarizarse con el HTML se pueden usar las herramientas de inspección del navegador ("Inspeccionar elemento"), esto les permitirá ver fácilmente la estructura de los elementos, clases y atributos asociados a los enlaces que necesitan. También se pueden buscar patrones en las clases CSS, Wikipedia suele tener clases CSS específicas para diferentes tipos de enlaces, como enlaces a subcategorías o enlaces interlinguales. Reconocer estos patrones ayudará a identificar rápidamente los elementos concretos.

Paquetes de Python interesantes:

- `requests`: Para hacer peticiones HTTP. Es ideal para enviar solicitudes GET y recibir las páginas HTML de Wikipedia que luego serán analizadas.

- `beautifulsoup4`: Librería para parsear HTML y extraer datos de páginas web. Facilita la búsqueda de elementos específicos en el HTML (como enlaces a artículos, subcategorías, etc.) y permite navegar por la estructura del documento de forma sencilla.

- `lxml` (opcional, pero recomendado): Es un parser de HTML que funciona junto con **BeautifulSoup**. Es más rápido que el parser estándar y puede manejar HTML mal formado mejor. Si hay problemas al parsear páginas, este paquete puede ayudar.

- **`matplotlib`** (útil para visualización): Se utiliza para crear gráficos y visualizaciones de datos, p.e.: gráficos de barras, circulares o diagramas de árbol para visualizar jerarquías de categorías o las comparaciones de artículos.

- **`networkx`** (para diagramas de categorías): Librería para crear, analizar y visualizar grafos. Puede ser útil para representar la jerarquía de categorías de Wikipedia como un grafo y mostrar cómo se relacionan las subcategorías entre sí.

- **`googletrans`**: Permite traducir texto usando la API de Google Translate. Puede ser muy útil para traducir automáticamente títulos de artículos o secciones, facilitando la comparación entre las versiones en español e inglés. (*Nota: puede haber limitaciones de uso o cambios en la API, usar con moderación para evitar bloqueos temporales*).

Paquete	Descripción	Instalación
`requests`	Para hacer peticiones HTTP y descargar HTML	`pip install requests`
`beautifulsoup4`	Parsear HTML y extraer datos	`pip install beautifulsoup4`
`lxml`	Parser avanzado de HTML para mejorar el scraping	`pip install lxml`
`googletrans`	Traducir textos automáticamente entre español e inglés	`pip install googletrans==4.0.0-rc1`
`matplotlib`	Crear gráficos y visualizaciones	`pip install matplotlib`
`networkx`	Representar y analizar grafos para visualizar jerarquías	`pip install networkx`