

2024 GUIA INSTRUCTIVA

Flores Espinoza Jose Enrique



Clonación de proyectos y Instalación de dependencias

Paso 1: Clonar el repositorio

```
git clone https://github.com/tokien736/CuscoTrends.git
```

- **Qué hace:** Este comando descarga una copia del repositorio de GitHub en tu máquina local. Git es una herramienta de control de versiones, y git clone te permite crear una copia de un proyecto que está almacenado en un servidor remoto (en este caso, GitHub). Aquí, estás clonando el proyecto llamado CuscoTrends.

Paso 2: Instalación de dependencias

Antes de trabajar con el proyecto, necesitas instalar las bibliotecas y dependencias necesarias para ejecutar el código. Esto se realiza utilizando pip, el administrador de paquetes para Python.

1. Instalación de requests:

```
pip install requests
```

- **Qué hace:** Requests es una biblioteca que permite hacer peticiones HTTP de manera sencilla. Es útil para obtener datos de sitios web o APIs y es fundamental en scraping web.

2. Instalación de BeautifulSoup4:

```
pip install beautifulsoup4
```

- **Qué hace:** BeautifulSoup es una biblioteca de Python utilizada para extraer datos de archivos HTML y XML. Se usa principalmente en scraping web para extraer información estructurada de páginas web.

3. Instalación de pandas:

```
pip install pandas
```

- **Qué hace:** Pandas es una biblioteca que facilita la manipulación y análisis de datos, especialmente con estructuras de datos tabulares como DataFrames. Es común en análisis de datos y machine learning.

4. Instalación de numpy:

```
pip install numpy
```

- **Qué hace:** Numpy es una biblioteca para realizar cálculos matemáticos eficientes en Python. Proporciona soporte para grandes arreglos y matrices multidimensionales, que son usados en matemáticas y cálculos científicos.

5. Instalación de matplotlib:

```
pip install matplotlib
```

- **Qué hace:** Matplotlib es una biblioteca que permite crear gráficos en 2D en Python. Es comúnmente usada para generar gráficos de líneas, histogramas, gráficos de dispersión, etc.

6. Instalación de seaborn:

```
pip install seaborn
```

- **Qué hace:** Seaborn es una biblioteca basada en Matplotlib, pero que proporciona una interfaz más fácil para hacer gráficos avanzados y atractivos. Es ideal para la visualización de datos estadísticos.

7. Instalación de scikit-learn:

`pip install scikit-learn`

- **Qué hace:** Scikit-learn es una biblioteca de machine learning que proporciona herramientas para modelado predictivo, clasificación, regresión, y clustering.

8. Instalación de urllib3:

`pip install urllib3`

- **Qué hace:** Urllib3 es una biblioteca que maneja conexiones HTTP, proporcionando control avanzado sobre las conexiones, incluyendo reintentos y conexiones seguras. Requests utiliza urllib3 de fondo.

Dependencias adicionales incluidas por defecto:

`concurrent.futures`: Ya está incluido en Python 3.2+. No necesitas instalarlo por separado, es parte de la librería estándar de Python.

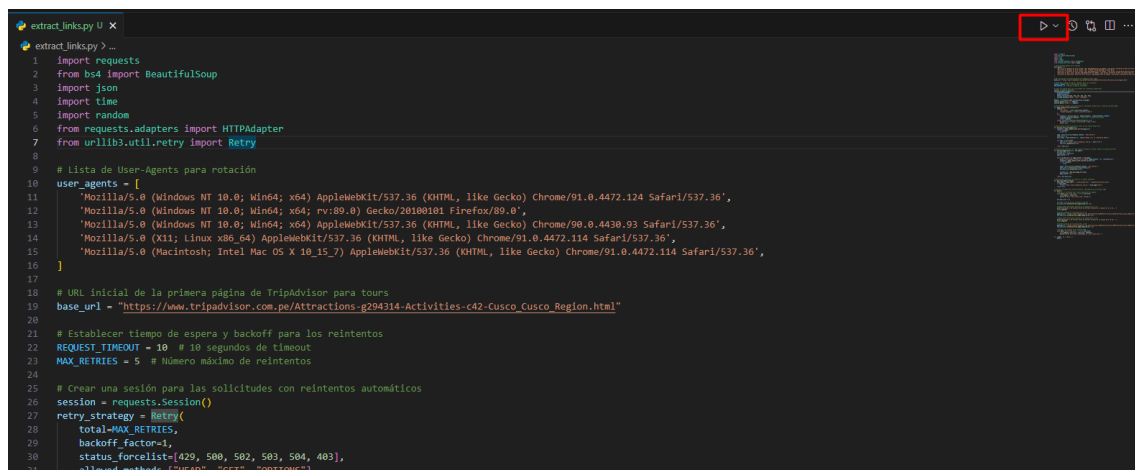
Dependencias opcionales:

- `os`:
Parte de la librería estándar de Python, usada para interactuar con el sistema operativo.
- `json`:
Biblioteca estándar para trabajar con archivos y datos en formato JSON.
- `random`:
Parte de la librería estándar para generar números y valores aleatorios.

Modo de Ejecución:

Paso 1:

Ejecutar el archivo llamado `extract_links.py`, el cual extraerá las URLs según la carpeta en la que se encuentre. Existen dos carpetas: **TripAdvisor** y **Trustpilot**.



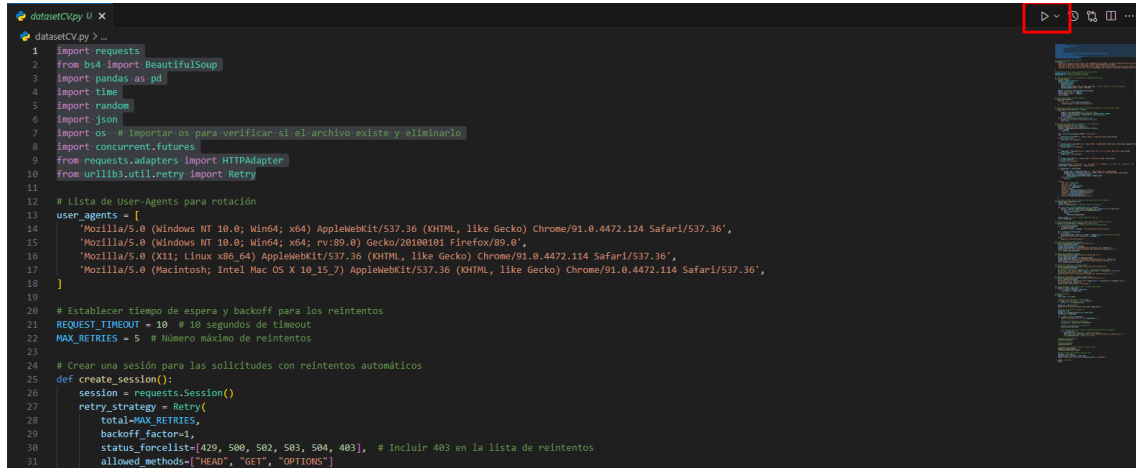
```

1 import requests
2 from bs4 import BeautifulSoup
3 import json
4 import time
5 import random
6 from requests.adapters import HTTPAdapter
7 from urllib3.util.retry import Retry
8
9 # Lista de User-Agents para rotación
10 user_agents = [
11     'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36',
12     'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0',
13     'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.93 Safari/537.36',
14     'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.114 Safari/537.36',
15     'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.114 Safari/537.36',
16 ]
17
18 # URL inicial de la primera página de TripAdvisor para tours
19 base_url = "https://www.tripadvisor.com.pe/Attractions-g294314-Activities-c42-Cusco_Cusco_Region.html"
20
21 # Establecer tiempo de espera y backoff para los reintentos
22 REQUEST_TIMEOUT = 10 # 10 segundos de timeout
23 MAX_RETRIES = 5 # Número máximo de reintentos
24
25 # Crear una sesión para las solicitudes con reintentos automáticos
26 session = requests.Session()
27 retry_strategy = Retry(
28     total=MAX_RETRIES,
29     backoff_factor=1,
30     status_forcelist=(429, 500, 502, 503, 504, 403),
31     allowed_methods=("HEAD", "GET", "OPTIONS")

```

Paso 2:

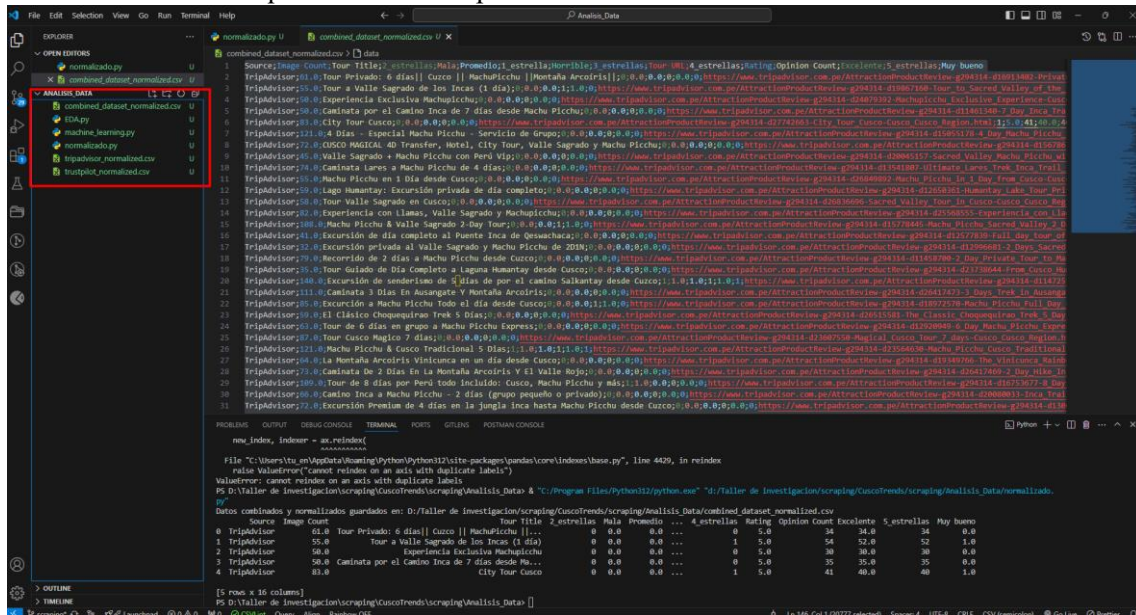
Ejecutar el archivo datasetCV.py, que está en ambas carpetas. Este archivo extrae los datos de cada página, previamente guardados por el script anterior en formato JSON, y navega por los archivos JSON.



```
1 import requests
2 from bs4 import BeautifulSoup
3 import pandas as pd
4 import time
5 import random
6 import json
7 import os # Importar os para verificar si el archivo existe y eliminarlo
8 import concurrent.futures
9 from requests.adapters import HTTPAdapter
10 from urllib3.util.retry import Retry
11
12 # Lista de User-Agents para rotación
13 user_agents = [
14     'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36',
15     'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0',
16     'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.114 Safari/537.36',
17     'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.114 Safari/537.36',
18 ]
19
20 # Establecer tiempo de espera y backoff para los reintentos
21 REQUEST_TIMEOUT = 10 # 10 segundos de timeout
22 MAX_RETRIES = 5 # Número máximo de reintentos
23
24 # Crear una sesión para las solicitudes con reintentos automáticos
25 def create_session():
26     session = requests.Session()
27     retry_strategy = Retry(
28         total=MAX_RETRIES,
29         backoff_factor=1,
30         status_forcelist=[429, 500, 502, 503, 504, 403], # Incluir 403 en la lista de reintentos
31         allowed_methods=["HEAD", "GET", "OPTIONS"]
32     )
33     session.mount("https://", HTTPAdapter(max_retries=retry_strategy))
34     return session
```

Paso 3:

Normalizar los datos que estan en la carpeta Analisis de datos



```
1 import pandas as pd
2
3 def normalize_data():
4     # Leer los datos desde el archivo CSV
5     data = pd.read_csv('combined_dataset_normalizado.csv')
6
7     # Procesar los datos
8     # ... (código de procesamiento de datos) ...
9
10    # Guardar los datos normalizados en un nuevo archivo CSV
11    data.to_csv('normalizado.csv', index=False)
```

Primero ejecutamos Normalizado.py luego las demas programmas