1. Approach to the Solution

Objective:

The script automates the extraction, text cleaning, and analysis of articles sourced from URLs listed in an Excel file (Input.xlsx). The primary goal is to perform sentiment analysis and readability assessment on the extracted text, providing structured output in an Excel format.

Solution Breakdown:

1.1. Environment Setup and Imports:

- The script begins by ensuring all necessary dependencies (beautifulsoup4, nltk, pandas, etc.) are installed using os.system to run pip commands programmatically.
- Essential Python libraries (pandas, requests, BeautifulSoup, nltk) are imported to handle data extraction, manipulation, and analysis.
- NLTK resources (punkt and stopwords) are downloaded to facilitate text tokenization and filtering.

1.2. Data Extraction from Excel:

- The user is prompted to provide the path to the Input.xlsx file, which contains a list of URLs.
- The file is read using pandas.read_excel(), and the extracted data is converted to CSV format for easier access and backup.
- If the Excel file is not found, the program gracefully exits with an error message, preventing further execution.

1.3. Web Scraping and Article Extraction:

- The script iterates through the URLs listed in the Excel file, fetching web pages using the requests library.
- BeautifulSoup is employed to parse the HTML and extract paragraph text (<p> tags), consolidating it into a single text block.
- Each article is saved as a .txt file in the extracted_articles directory, with the filename corresponding to the unique URL_ID from the Excel sheet.

1.4. Text Cleaning and Stopwords Removal:

- The script dynamically prompts the user for paths to various stopword lists (e.g., StopWords_Auditor.txt, StopWords_Generic.txt).
- Binary files (e.g., StopWords_Currencies.txt) are handled by reading the content in binary mode and decoding with utf-8, replacing problematic characters if necessary.
- A union of all stopword sets is created to filter unwanted words from the extracted articles.

1.5. Sentiment and Readability Analysis:

- Paths to sentiment word lists (positive-words.txt, negative-words.txt) are requested from the user.

- The text from each article is tokenized using NLTK's word_tokenize and sent_tokenize methods.
- The script calculates the following metrics:
    - **Sentiment Scores:** Positive, negative, and polarity scores.
    - **Readability Metrics:** Flesch Reading Ease, fog index, and percentage of complex words.
    - **Additional Analysis:** Average sentence length, syllables per word, and personal pronoun count.
- To avoid division by zero errors, the script includes condition checks for empty or incomplete text data.

1.6. Result Compilation and Output to Excel:

- The analysis results for each article are consolidated into a pandas DataFrame.
- The results are saved as Output Data Structure.xlsx in the user-specified directory.
- The program provides a completion message, confirming the successful generation of the output file.

---

2. How to Run the Python File

Prerequisites:

Ensure the following tools are installed:

- **Python 3.8 or higher**
- **Pip (Python Package Manager)**

Running the Script:

1. **Open Command Prompt or Terminal**
2. **Navigate to the Script Directory:**
3. cd path/to/your/script/directory
4. **Run the Python Script:**
5. python Text_Extraction_and_Analysis.py

Execution Flow:

- The script will sequentially prompt for:
    - Path to Input.xlsx (containing URLs)
    - Path to save the CSV output (Input.csv)
    - Paths to various stopword files
    - Paths to sentiment analysis dictionaries

  - Path to save the final output Excel file

- If the user presses Enter without providing a path, the script defaults to the current directory (os.getcwd()).

Example Run:

```
Enter the path to Input.xlsx (or press Enter for default): C:/Users/John/Documents/Input.xlsx

Enter the path to save Input.csv (or press Enter for default):

CSV saved to: Input.csv

Enter path to StopWords_Auditor.txt: C:/Users/John/StopWords/StopWords_Auditor.txt

Enter path to StopWords_Currencies.txt (Binary): C:/Users/John/StopWords/StopWords_Currencies.txt

Enter path to positive-words.txt: C:/Users/John/Dictionaries/positive-words.txt

Enter path to negative-words.txt: C:/Users/John/Dictionaries/negative-words.txt

Results saved to Output Data Structure.xlsx
```

---

3. Dependencies Required

The following dependencies are essential for successful script execution:

Core Python Libraries:

```
pip install pandas nltk beautifulsoup4 requests syllapy openpyxl setuptools
```

Required NLTK Packages:

Ensure the following resources are downloaded:

```
import nltk

nltk.download('punkt')

nltk.download('stopwords')
```

Dependency List:

- pandas – Excel/CSV file manipulation.
- nltk – Tokenization and text processing.
- beautifulsoup4 – Web scraping and HTML parsing.
- requests – Fetching content from URLs.
- syllapy – Syllable counting for readability metrics.
- openpyxl – Excel file creation and editing.

---

4. Error Handling and Robustness:

- **File Not Found:** If the provided Excel file or stopword list is not found, the script will alert the user and gracefully exit.
- **Invalid URLs:** Any URL that cannot be accessed will be skipped, with an error message displayed.
- **Encoding Issues:** Binary files or non-UTF-8 encoded stopword lists are handled using errors='replace'.
- **Division by Zero:** Empty articles or incomplete data are skipped to prevent division errors during readability analysis.

---

5. Output:

- **CSV Conversion:** The Excel file (Input.xlsx) is converted to Input.csv.
- **Extracted Articles:** Articles are saved as .txt files in the extracted_articles directory.
- **Final Output:** Sentiment and readability analysis results are compiled into Output Data Structure.xlsx.
