Text Extraction and Analysis

Approach to the Solution

The task required extracting article texts from given URLs and performing various textual analyses to generate specific metrics. Here's a detailed breakdown of how I approached the solution:

Text Extraction:

- Used **Selenium** for web scraping to handle dynamic content that might not be easily accessible via simple HTTP requests.
- Configured Selenium to run in headless mode to ensure it can run without a graphical interface.
- Extracted article titles and texts, ensuring we avoid unwanted content like headers and footers.

Data Cleaning:

- Used NLTK to tokenize text and remove stop words.
- Ensured only relevant tokens (words) were kept for analysis.

Sentiment and Readability Analysis:

- Defined functions to calculate various metrics like positive/negative scores, polarity, subjectivity, sentence lengths, complex word counts, and more.
- Calculated readability metrics using the Gunning Fog Index and other related formulas.

Handling Input and Output:

- Read input data from an Excel file.
- Prompted the user to either analyze a specific URL by its ID or all URLs.
- Compiled the results into a **data frame** and saved it to an output Excel file, ensuring the structure matched the required format.

Error Handling and Robustness:

- Implemented error handling to manage issues like missing files, empty texts, and division by zero.
- Ensured the script could handle partial failures (e.g., if some URLs failed to load) without crashing.

How to Run the Script

To run the script and generate the output, follow these steps:

```
pip install pandas nltk selenium webdriver-manager openpyxl
```

Download NLTK Data: Run the following commands in your Python environment to download the necessary NLTK data:

```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
```

Set Up Directory Structure: Ensure you have the following directories and files:

StopWords: Contains stop word lists.
MasterDictionary: Contains positive and negative word lists.
Input.xlsx: The input file with URLs to analyze.
Output Data Structure.xlsx: The output file where results will be saved.

IMPORTANT:
MAKE SURE Output Data Structure.xlsx is empty
The script will prompt you to enter the URL_ID to analyze (or type 'All' to analyze all URLs). After processing, the results will be saved to Output Data Structure.xlsx.

Dependencies

Here is a list of dependencies required to run the script:

- pandas: For data manipulation and reading/writing Excel files.
- nltk: For natural language processing tasks.
- selenium: For web scraping.
- webdriver-manager: To manage browser drivers for Selenium.
- openpyxl: For Excel file operations.
- Ensure you have Python installed. Then, install the required packages using pip