Explanation of Approach

1. **Fetch All Article Titles and Text**:
    - First, I read the URLs from the Input.xlsx file to fetch the articles.
    - I used the requests library to send HTTP GET requests to these URLs and retrieve the HTML content.
    - The BeautifulSoup library was used to parse the HTML and extract the title and main text of each article, while removing the footer text using a predefined regex pattern.

2. **Save Original and Cleaned Articles**:
    - I removed stopwords from the article text using a list of stopwords located in the stopwords directory. The cleaned article text was then saved to file named with the URL ID.

3. **Calculate Variables**:
    - **Positive and Negative Scores**: I tokenized the cleaned article text and compared the tokens against a master dictionary of positive and negative words (excluding stopwords).
    - **Polarity Score**: Calculated using the formula (positive_score - negative_score) / ((positive_score + negative_score) + 0.000001).
    - **Subjectivity Score**: Calculated as (positive_score + negative_score) / (number of tokens + 0.000001).

    - **Readability Metrics**:
        - **Average Sentence Length**: Calculated as the total number of words divided by the total number of sentences.
        - **Percentage of Complex Words**: Calculated as the number of complex words divided by the total number of words.
        - **Fog Index**: Calculated as 0.4 * (average_sentence_length + percentage_of_complex_words).
        - **Syllable Count Per Word**: Calculated as the total number of syllables divided by the total number of words.
        - **Personal Pronouns**: Counted using a regex search for specific personal pronouns.
        - **Average Word Length**: Calculated as the total number of characters in all words divided by the total number of words.

4. **Store Results**:
    - I collected all calculated variables and additional information (URL ID and URL) in a dictionary.
    - These dictionaries were appended to a list, which was then used to create a Pandas DataFrame.

5. **Export Results to Excel**:
    - The final DataFrame, containing all calculated variables for each article, was exported to an Excel file (Output.xlsx).

Instructions for Running the Python Script

Ensure you have the following files in the same directory:

- text_analysis.py
- Input.xlsx
- stopwords directory (containing stopword text files)
- master directory (containing positive-words.txt and negative-words.txt)

Run the Script:

- Open a terminal or command prompt.
- Navigate to the directory containing your script and files.
- Execute the script using Python : python text_analysis.py

Output:

- The script will process the articles and generate two output files:
    - Cleaned article text files in the articles directory.
    - An Excel file named Output.csv containing the calculated variables.

Dependencies

Install required Python packages :

- RUN THE CODE : pip install pandas requests beautifulsoup4 nltk openpyxl

pandas: For data manipulation and analysis.

requests: For making HTTP requests to fetch web pages.

beautifulsoup4: For parsing HTML content.

nltk: For natural language processing tasks.

openpyxl: For reading and writing Excel files.