**README: Text Analysis Script**

**Overview**

This script is designed to process a list of URLs from an input Excel file, extract and clean the text from the web pages, perform sentiment and readability analysis, and output the results in a structured CSV format. It also saves the extracted articles in a separate folder.

**Features**

1. **Text Extraction:**
   * Extracts titles and paragraph content from web pages.
   * Saves the extracted text in separate .txt files for each URL.
2. **Text Cleaning:**
   * Removes stop words using a combination of:
     + Default NLTK stop words.
     + Custom stop words from files in the Stop Words folder.
   * Removes extra whitespace and normalizes text.
3. **Sentiment Analysis:**
   * Calculates positive and negative scores based on custom dictionaries.
   * Computes polarity and subjectivity scores.
4. **Readability Analysis:**
   * Computes average sentence length, Fog Index, and word-level metrics.
   * Counts complex words, syllables per word, and average word length.
5. **Error Handling:**
   * Catches and logs errors during URL processing or file reading.
   * Ensures results are saved even if some URLs fail to process.

**Dependencies**

The following Python libraries are required:

* pandas
* requests
* BeautifulSoup (from bs4)
* re (built-in)
* nltk
* textstat
* textblob
* os (built-in)

To install missing libraries:

pip install pandas requests beautifulsoup4 nltk textstat textblob

**Folder and File Requirements**

1. **Input Files:**
   * Input.xlsx: Contains two columns:
     + URL\_ID: A unique identifier for each URL.
     + URL: The web page address to process.
   * Output Data Structure.xlsx: Template file to structure the final output.
2. **Stop Words Folder:**
   * Folder named Stop Words containing multiple .txt files with additional stop words.
3. **Master Dictionary:**
   * positive-words.txt: Contains a list of positive words.
   * negative-words.txt: Contains a list of negative words.
4. **Output Files:**
   * output\_results.csv: Final analysis results.
   * extracted\_articles/: Folder where extracted articles are saved as .txt files.

**How to Use**

1. **Prepare the Input Files:**
   * Place Input.xlsx, Output Data Structure.xlsx, and the Stop Words folder in the same directory as the script.
   * Ensure the Master Dictionary folder with positive-words.txt and negative-words.txt is available.
2. **Run the Script:**
3. python script\_name.py
4. **Check the Results:**
   * Processed results are saved in output\_results.csv.
   * Extracted articles are saved in the extracted\_articles/ folder.

**Output Metrics**

1. **Sentiment Metrics:**
   * POSITIVE SCORE: Count of positive words.
   * NEGATIVE SCORE: Count of negative words.
   * POLARITY SCORE: Indicates overall sentiment (-1 to 1).
   * SUBJECTIVITY SCORE: Indicates objectivity/subjectivity (0 to 1).
2. **Readability Metrics:**
   * AVG SENTENCE LENGTH: Average words per sentence.
   * FOG INDEX: Readability metric.
   * WORD COUNT: Total words in the cleaned text.
   * COMPLEX WORD COUNT: Count of words with ≥3 vowels.
   * AVG WORD LENGTH: Average length of words.
   * SYLLABLE PER WORD: Average syllables per word.

**Error Handling**

* If any URL fails, the error is logged, and default values (0) are assigned to metrics for that URL.

**Troubleshooting**

1. **Missing Libraries:**
   * Install missing dependencies using pip install.
2. **Folder/File Not Found:**
   * Ensure required files and folders are correctly named and in the same directory.
3. **Timeouts:**
   * Increase the timeout value in the requests.get call if URLs take longer to load:
   * response = requests.get(url, timeout=20)
4. **Invalid URLs:**
   * Ensure all URLs in Input.xlsx are valid and accessible.