**Instructions for Data Extraction and NLP Test Assignment**

**Objective**

The objective of this assignment is to extract textual data from articles, perform text analysis, and compute several variables as specified. The extracted article text will be saved in text files, and the computed variables will be saved in an Excel file following a predefined structure.

**Approach**

1. **Data Extraction**:

- Extract article text from each URL provided in the `Input.xlsx` file.

- Only the article title and the article text are extracted, ignoring headers, footers, and other non-article content.

**Libraries Used:**

* **requests**: Used to fetch the HTML content from URLs.
* **BeautifulSoup**: Used for parsing HTML and extracting specific content (like article title and text).

**Process:**

* The script reads URLs from the Input.xlsx file using pandas.
* For each URL:
  + It sends a GET request to fetch the web page content.
  + Uses BeautifulSoup to parse the HTML and extract:
    - Article title (<h1> tag content).
    - Article text (<p> tags content).
  + Concatenates all <p> tags content into a single string to form the article text.

**Implementation:**

* The extract\_article\_text(url) function handles this process.
* It returns the extracted title and article text.

2. **Text Analysis:**

- Perform textual analysis on each extracted article to compute the following variables:

- POSITIVE SCORE

- NEGATIVE SCORE

- POLARITY SCORE

- SUBJECTIVITY SCORE

- AVG SENTENCE LENGTH

- PERCENTAGE OF COMPLEX WORDS

- FOG INDEX

- AVG NUMBER OF WORDS PER SENTENCE

- COMPLEX WORD COUNT

- WORD COUNT

- SYLLABLE PER WORD

- PERSONAL PRONOUNS

- AVG WORD LENGTH

**Libraries Used:**

* **nltk**: Used for natural language processing tasks such as tokenization and sentence splitting.

### **Text Analysis Functions**

1. **Positive Score (positive\_score(text)):**
   * Tokenizes the text into words.
   * Counts the number of words that appear in the positive-words.txt dictionary.
2. **Negative Score (negative\_score(text)):**
   * Similar to positive\_score, but counts words in the negative-words.txt dictionary.
3. **Polarity Score (polarity\_score(pos\_score, neg\_score)):**
   * Computes the polarity score using the formula:Polarity Score=Positive Score−Negative ScorePositive Score+Negative Score+0.000001Polarity Score=Positive Score+Negative Score+0.000001Positive Score−Negative Score​
   * Handles edge cases where scores might be zero.
4. **Subjectivity Score (subjectivity\_score(pos\_score, neg\_score, total\_words)):**
   * Computes the subjectivity score using the formula:Subjectivity Score=Positive Score+Negative ScoreTotal Words+0.000001Subjectivity Score=Total Words+0.000001Positive Score+Negative Score​
5. **Average Sentence Length (avg\_sentence\_length(text)):**
   * Splits text into sentences and calculates the average number of words per sentence.
6. **Percentage of Complex Words (percentage\_of\_complex\_words(text)):**
   * Counts words with three or more syllables and calculates their percentage in the total words.
7. **FOG Index (fog\_index(text)):**
   * Computes the FOG index using:FOG Index=0.4×(Average Sentence Length+Percentage of Complex Words)FOG Index=0.4×(Average Sentence Length+Percentage of Complex Words)
8. **Average Number of Words per Sentence (avg\_words\_per\_sentence(text)):**
   * Computes the total number of words divided by the number of sentences.
9. **Complex Word Count (complex\_word\_count(text)):**
   * Counts words with three or more syllables.
10. **Word Count (word\_count(text)):**
    * Counts total words in the text.
11. **Syllables per Word (syllables\_per\_word(text)):**
    * Computes the average number of syllables per word.
12. **Personal Pronouns (personal\_pronouns(text)):**
    * Counts occurrences of personal pronouns ('I', 'we', 'my', 'ours', 'us').
13. **Average Word Length (avg\_word\_length(text)):**
    * Computes the average length of words in characters.

3. **Output:**

- Save the results in an Excel file (`Output Data Structure.xlsx`) in the required format.

### **Summary**

* Data Extraction: Uses requests to fetch HTML and BeautifulSoup to extract article content.
* Text Analysis: Utilizes nltk for tokenization and sentence splitting, and custom logic for computing metrics like polarity, subjectivity, average sentence length, and more.

This structured approach ensures accurate extraction and analysis of textual data from web articles, fulfilling the requirements specified in the assignment.

**Dependencies**

Ensure you have the following Python packages installed:

- pandas

- requests

- beautifulsoup4

- nltk

- openpyxl

You can install these packages using the following command:

***pip install pandas requests beautifulsoup4 nltk openpyxl***

**Files**

Ensure you have the following files in the same directory as the script:

- `Input.xlsx`: The input file containing URLs.

- `MasterDictionary/positive-words.txt`: A text file containing positive words.

- `MasterDictionary/negative-words.txt`: A text file containing negative words.

- `article\_analysis.py`: The Python script file.

**Running the Script**

1. **Download NLTK data:**

- The script uses the NLTK library, which requires some initial setup to download necessary datasets. This is handled within the script itself by using `nltk.download('punkt')`.

2. **Run the Script:**

- Open a terminal or command prompt.

- Navigate to the directory containing the script and the necessary files.

- Execute the script by running:

***python article\_analysis.py***

3. **Output:**

- The script will generate an `Output Data Structure.xlsx` file containing the results of the text analysis.

**Explanation of the Script**

1. **Loading Positive and Negative Words:**

- Positive and negative words are loaded from the respective text files using `ISO-8859-1` encoding to avoid decoding issues.

2. **Reading Input File:**

- The `Input.xlsx` file is read using pandas to get the list of URLs.

3. **Text Extraction:**

- For each URL, the script uses the `requests` library to fetch the web page content and `BeautifulSoup` to parse and extract the article title and text.

4. **Text Analysis Functions:**

- Several functions are defined to compute the required variables, including positive and negative scores, polarity score, subjectivity score, and more.

5. **Processing URLs:**

- The script processes each URL, extracts the text, performs text analysis, and computes the required variables.

6. **Saving Results:**

- The results are saved to an `Output Data Structure.xlsx` file in the required format.