**BLACKCOFFER**

**Text analysis**

This Python script is designed to automatically fetch articles from a list of URLs, scrape the content, and perform textual analysis on each article. The analysis results are saved both in individual text files for each article and in a summary Excel sheet that contains various metrics derived from the articles. The purpose of this script is to help analyze large volumes of text quickly and extract valuable insights such as readability, sentiment, and word usage patterns.

**Automated Article Scraping:**

The script reads an input Excel file (Input.xlsx) containing a list of article URLs and their corresponding identifiers (URL\_ID).

For each URL, the script fetches the HTML content of the article and extracts the article’s title and body text. It saves this information in a separate text file named after the URL\_ID.

**Textual Analysis Metrics:**

The script performs several types of textual analysis on each article to provide insights into readability, sentiment, and word usage:

**Readability Metrics:**

* **Word Count**: Total number of words in the article.
* **Character Count**: Total number of characters in the article.
* **Sentence Count**: Number of sentences, determined based on punctuation marks (periods, exclamation marks, and question marks).
* **Average Sentence Length**: Average number of words per sentence.
* **Percentage of Complex Words**: Percentage of words in the article that have more than two syllables.
* **Fog Index**: A readability score that indicates the complexity of the article based on sentence length and the percentage of complex words.

**Word and Syllable Analysis:**

* **Syllables per Word**: Average number of syllables per word.
* **Complex Word Count**: The number of complex words in the article (defined as words with more than two syllables).
* **Average Word Length**: The average length of words in terms of character count.

**Sentiment Analysis:**

* **Polarity Score**: Measures the sentiment of the article, with a range from -1 (negative) to 1 (positive).
* **Subjectivity Score**: Measures how subjective or objective the article is, with a range from 0 (objective) to 1 (subjective).
* **Positive and Negative Scores**: Counts of positive and negative sentiment words based on TextBlob analysis.

**Output Files:**

**Text Files**: Each article is saved as a .txt file in the articles directory, named after the URL\_ID. This file contains the article’s title and body text.

**Excel Summary File**: The analysis results for all articles are saved in an Excel file (output\_analysis.xlsx). This file contains a row for each article, with columns representing the various textual analysis metrics. It allows HR to review the overall sentiment and readability of multiple articles at a glance.

**Error Handling:**

If the script encounters any issues while processing an article (e.g., invalid URL, missing content), it logs the error and continues with the next article without halting the entire process. This ensures that the script runs smoothly even if some articles have issues.

**How to Use:**

**Prepare the Input File:**

The input Excel file (Input.xlsx) should contain two columns:

* + - **URL\_ID**: A unique identifier for each article.
    - **URL**: The URL of the article to scrape.

**Run the Script:**

Ensure the required Python libraries are installed (e.g., pandas, requests, beautifulsoup4, textblob).

The script will automatically fetch the articles, analyze the text, and save the results.

**View Results:**

The articles will be saved as text files in the articles directory.

The analysis results will be saved in an Excel file (output\_analysis.xlsx), which will include the following columns:

* + - **URL\_ID**: Unique identifier for each article.
    - **Title**: Title of the article.
    - **Positive Score**: Number of positive sentiment words.
    - **Negative Score**: Number of negative sentiment words.
    - **Polarity Score**: Overall sentiment of the article.
    - **Subjectivity Score**: Subjectivity of the article.
    - **Average Sentence Length**: Average number of words per sentence.
    - **Percentage of Complex Words**: Percentage of complex words.
    - **Fog Index**: Readability score.
    - **Complex Word Count**: Number of complex words.
    - **Word Count**: Total word count.
    - **Syllables per Word**: Average syllables per word.
    - **Personal Pronouns**: Count of personal pronouns.
    - **Average Word Length**: Average length of words

To run your script successfully, you'll need the following Python dependencies:

1. **pandas**: For handling data frames and reading/writing Excel files.
2. **requests**: To fetch content from URLs.
3. **BeautifulSoup** (from bs4): To parse and extract HTML content from the fetched pages.
4. **re**: For regular expressions used in text processing.
5. **textblob**: For performing sentiment analysis and polarity scoring.
6. **openpyxl** is also needed for reading and writing .xlsx files, which pandas will use when handling Excel files.

Here is the full list of dependencies you'll need to install:

**pip install pandas requests beautifulsoup4 textblob openpyxl**

Once these dependencies are installed, your script should work as expected

**About:**

A.Ashiq sheriff

Sheriffashiq7@gmail.com

9042555770