**BLACKCOFFER TASK**

**Objective:**

The objective of this assignment is to extract textual data articles from the given URL and perform text analysis to compute variables that are explained below.

1. POSITIVE SCORE
2. NEGATIVE SCORE
3. POLARITY SCORE
4. SUBJECTIVITY SCORE
5. AVG SENTENCE LENGTH
6. PERCENTAGE OF COMPLEX WORDS
7. FOG INDEX
8. AVG NUMBER OF WORDS PER SENTENCE
9. COMPLEX WORD COUNT
10. WORD COUNT
11. SYLLABLE PER WORD
12. PERSONAL PRONOUNS
13. AVG WORD LENGTH

**Methodology:**

This task begins with preparing the environment and setting up the necessary tools for text analysis. We start by downloading the necessary input file from Google Drive and opening it in Jupyter Notebook. The first step involves importing essential libraries such as pandas, numpy, os, re, requests, urllib, BeautifulSoup, and NLTK to facilitate text processing and analysis.

**1. Library Imports and Initial Setup:**

The WordNetLemmatizer from NLTK is initialized, and necessary NLTK data, including tokenizers, stopwords, and wordnet, are downloaded. The stopwords are a predefined set of words (like 'and', 'the', 'is', etc.) that are usually filtered out during text processing. These stopwords are imported to help clean the text data.

**2. Data Loading:**

A DataFrame is read from an Excel file containing the URLs that need to be analyzed. This DataFrame contains approximately 150 URLs. Alongside, various stopword lists and positive/negative word lists are loaded from specified file paths. These lists are essential for performing sentiment analysis and removing irrelevant words from the text data.

**3. Text Processing Function:**

A text processing function, text\_corpus, is defined. This function takes the raw text as input and performs several transformations:

* **Lowercasing:** Converts the entire text to lowercase to ensure uniformity.
* **Removing Non-Alphabetic Characters:** Uses regular expressions to remove any character that is not a letter.
* **Tokenization:** Breaks down the text into individual words using word\_tokenize.
* **Stopwords Removal:** Filters out stopwords to retain only meaningful words.
* **Lemmatization:** Converts words to their base forms (e.g., 'running' to 'run') using the WordNetLemmatizer.

This process ensures the text is clean, standardized, and ready for analysis.

**4. Sentiment Analysis:**

Sentiment analysis involves using predefined lists of positive and negative words. For each article, the following steps are performed:

* **Counting Positive and Negative Words:** The function checks each word in the processed text against the positive and negative word lists, counting the occurrences of each.
* **Calculating Polarity Score:** The polarity score is calculated using the formula: Polarity Score=(Positive Score−Negative Score)(Positive Score+Negative Score)+0.000001\text{Polarity Score} = \frac{(\text{Positive Score} - \text{Negative Score})}{(\text{Positive Score} + \text{Negative Score}) + 0.000001}Polarity Score=(Positive Score+Negative Score)+0.000001(Positive Score−Negative Score)​ This score indicates whether the overall sentiment of the text is positive or negative.
* **Calculating Subjectivity Score:** The subjectivity score is calculated as: Subjectivity Score=(Positive Score+Negative Score)(Total Words after cleaning)+0.000001\text{Subjectivity Score} = \frac{(\text{Positive Score} + \text{Negative Score})}{(\text{Total Words after cleaning}) + 0.000001}Subjectivity Score=(Total Words after cleaning)+0.000001(Positive Score+Negative Score)​ This score indicates the extent to which the text is subjective or objective.

**5. Sentence and Word Counting:**

To understand the readability and structure of the text, the total number of words and sentences in each article is counted:

* **Word Count:** The total number of words is counted using word\_tokenize.
* **Sentence Count:** The total number of sentences is counted using sent\_tokenize.
* **Average Sentence Length:** The average sentence length is calculated as the total number of words divided by the number of sentences.

**6. Syllable Counting:**

A function, count\_syllables, is defined to count the number of syllables in each word. This function:

* Initializes a syllable count to 0.
* Checks for vowels in each word and increments the count accordingly.
* Makes adjustments for specific endings like 'es', 'ed', and 'e' to accurately count syllables.
* Returns the total syllable count for each word.

**7. Complex Words and FOG Index:**

The number of complex words (words with more than two syllables) is determined. Using this, the percentage of complex words is calculated as:

Percentage of Complex Words=Number of Complex WordsTotal Number of Words\text{Percentage of Complex Words} = \frac{\text{Number of Complex Words}}{\text{Total Number of Words}}Percentage of Complex Words=Total Number of WordsNumber of Complex Words​

The FOG Index, which is a measure of text readability, is then calculated using the formula:

FOG Index=0.4×(Average Sentence Length+Percentage of Complex Words)\text{FOG Index} = 0.4 \times (\text{Average Sentence Length} + \text{Percentage of Complex Words})FOG Index=0.4×(Average Sentence Length+Percentage of Complex Words)

**8. Average Word Length:**

The average word length is calculated by dividing the sum of the total number of characters in each word by the total number of words. This provides insights into the complexity of the vocabulary used in the text.

**9. Personal Pronoun Counting:**

A function, count\_personal\_pronouns, is defined to count personal pronouns. The function uses a regular expression pattern to match words like "I", "we", "my", "ours", and "us". Special care is taken to exclude matches for the country name "US". The result is a list of all personal pronouns found in the text.

**10. Result Compilation and Saving:**

For each URL, the extracted and calculated data, including sentiment scores, average sentence length, syllable counts, complex word counts, FOG Index, average word length, and personal pronoun counts, are appended to a list. This list is then converted into a DataFrame. The DataFrame is saved to a CSV file, which contains all the results for further analysis.

This comprehensive methodology ensures a thorough analysis of the textual data, providing valuable insights into the sentiment, readability, and complexity of the articles. The steps are methodically executed to transform raw text into meaningful metrics that can be used for various analytical purposes.

Top of Form

**HOW TO RUN THE PY FILE:**

**Set Up Your Environment:**

* Ensure you have Python installed on your system. You can download it from the [official Python website](https://www.python.org/downloads/).
* Open your terminal (or command prompt) and navigate to the directory where your .py file is located.
* Install the required libraries using pip. You can do this manually or create a requirements.txt file and use it to install all dependencies at once.

**Dependencies:**

Pandas

numpy

Urllib

re(regular expression)

Requests

beautifulsoup4

nltk

Ensure File Paths Are Correct:

* Make sure the file paths in your script (e.g., for reading the Excel file and stopwords lists) are correct and accessible.

Bottom of Form