**Project Overview**

This project involved the extraction of textual data from specified URLs and the analysis of this text to compute various readability and sentiment metrics. The final objective was to process these texts, analyze them for various linguistic features, and output the results in a CSV file. This detailed report explains the entire procedure, including the tools used, the methodologies applied, and the steps taken to complete the project.

**1. Project Objectives**

**Primary Objective:**

* Extract articles from URLs provided in an input file.
* Analyze the extracted text for sentiment and readability.
* Generate and save various metrics into a CSV file.

**2. Tools and Libraries Used**

**Libraries and Tools:**

* **Requests:** To send HTTP requests and fetch web page contents.
* **BeautifulSoup:** For parsing HTML content and extracting the relevant article text.
* **TextBlob:** For sentiment analysis, including calculating polarity and subjectivity.
* **NLTK (Natural Language Toolkit):** For tokenizing text into sentences and words.
* **Textstat:** For computing readability metrics like the FOG Index, percentage of complex words, and syllable count.

**3. Procedure**

**1. Data Extraction**

**Objective:** Extract the text of articles from given URLs while ignoring irrelevant parts of the web page (headers, footers, ads).

**Steps:**

1. **Reading Input Data:**
   * The URLs were provided in an Input.xlsx file, which was read using the pandas library.
2. **Fetching Web Pages:**
   * Utilized the requests library to fetch HTML content from each URL.
   * Example:

python

Copy code

import requests

response = requests.get(url)

html\_content = response.text

1. **Parsing HTML:**
   * Used BeautifulSoup to parse the HTML and extract only the article text.
   * Example:

python

Copy code

from bs4 import BeautifulSoup

soup = BeautifulSoup(html\_content, 'html.parser')

article\_text = soup.find(class\_='article-content').get\_text()

1. **Saving Extracted Text:**
   * The extracted article text was saved to a text file with a filename based on the URL ID.

**2. Text Analysis**

**Objective:** Analyze the text to compute various metrics including sentiment scores, readability indices, and text complexity.

**Steps:**

1. **Sentiment Analysis:**
   * Applied TextBlob to compute polarity and subjectivity scores.
   * Example:

python

Copy code

from textblob import TextBlob

blob = TextBlob(article\_text)

polarity = blob.sentiment.polarity

subjectivity = blob.sentiment.subjectivity

1. **Readability Metrics:**
   * **Average Sentence Length:**

python

Copy code

import nltk

sentences = nltk.sent\_tokenize(article\_text)

avg\_sentence\_length = sum(len(sentence.split()) for sentence in sentences) / len(sentences)

* + **Percentage of Complex Words:**

python

Copy code

import textstat

perc\_complex\_words = textstat.percentage\_of\_complex\_words(article\_text)

* + **FOG Index:**

python

Copy code

fog\_index\_score = textstat.gunning\_fog(article\_text)

* + **Average Words Per Sentence:**

python

Copy code

avg\_words\_per\_sentence = sum(len(sentence.split()) for sentence in sentences) / len(sentences)

* + **Complex Word Count:**

python

Copy code

complex\_word\_count = textstat.complex\_word\_count(article\_text)

* + **Word Count:**

python

Copy code

word\_count = len(article\_text.split())

* + **Syllables Per Word:**

python

Copy code

syllables\_per\_word = textstat.syllable\_count(article\_text) / len(article\_text.split())

* + **Personal Pronouns:**

python

Copy code

pronouns = ['I', 'me', 'my', 'mine', 'we', 'us', 'our', 'ours', 'you', 'your', 'yours', 'he', 'him', 'his', 'she', 'her', 'hers', 'they', 'them', 'their', 'theirs']

personal\_pronouns\_count = sum(word.lower() in pronouns for word in article\_text.split())

* + **Average Word Length:**

python

Copy code

avg\_word\_length = sum(len(word) for word in article\_text.split()) / len(article\_text.split())

1. **Generating the Output:**
   * Compiled all computed metrics into a structured format.
   * Saved the results in a CSV file using pandas.
   * Example:

python

Copy code

import pandas as pd

results = [[filename, polarity, subjectivity, avg\_sentence\_length, perc\_complex\_words, fog\_index\_score,

avg\_words\_per\_sentence, complex\_word\_count, word\_count, syllables\_per\_word,

personal\_pronouns, avg\_word\_length]]

df = pd.DataFrame(results, columns=['File Name', 'Polarity Score', 'Subjectivity Score', 'Avg Sentence Length',

'Percentage of Complex Words', 'Fog Index', 'Avg Words per Sentence',

'Complex Word Count', 'Word Count', 'Syllables per Word', 'Personal Pronouns',

'Avg Word Length'])

df.to\_csv('output.csv', index=False)

**4. Submission Preparation**

**Files Included:**

1. **Python Script (.py):** Contains all the code for data extraction and analysis.
2. **Output File (output.csv):** The CSV file with the computed metrics.
3. **Instruction File:** Detailed instructions on how to run the Python script and interpret the results.

**Instruction File Content:**

* **Approach to Solution:** Explanation of the process from data extraction to analysis.
* **Running the Script:** Instructions on how to execute the Python script.
* **Dependencies:** List of required libraries and how to install them.

**5. Challenges and Solutions**

**Challenges:**

* Ensuring accurate extraction of article text from diverse HTML structures.
* Handling large volumes of text data efficiently.
* Computing various readability and complexity metrics accurately.

**Solutions:**

* Used robust HTML parsing techniques to isolate article content.
* Leveraged efficient text processing libraries to manage and analyze data.
* Validated the output against known benchmarks to ensure accuracy.

**6. Final Notes**

Throughout the project, I utilized resources such as YouTube tutorials and ChatGPT to understand specific functions and best practices. This approach ensured that I followed effective methodologies and adhered to best practices in Python programming and text analysis.