Project title: News Aggregator and Analyzer

Submitted by:

Tanvin kabir Thuin

Roll: 27

Course: Basic Python

Enhancing Digital Government and Economy (EDGE)

University of Barishal

Submitted to:

Md. Rashid Al Asif

Assistant Professor Dept. of Computer Science & Engineering

University of Barishal

Date of Submission: 23 February 2025

Abstract

The News Aggregator and Analyzer project is a Python-based application designed to collect, categorize, analyze, and visualize news articles from various sources. This project incorporates fundamental programming concepts such as variables, data types, casting, strings, operators, lists, tuples, sets, dictionaries, loops, and functions.

The project begins by fetching news articles using web scraping techniques and then categorizes them into different topics like politics, sports, entertainment, and technology. It further processes the text to extract keywords and perform sentiment analysis using natural language processing (NLP) libraries. Additionally, the application visualizes the sentiment analysis results and keyword trends through graphical representations.

By integrating these features, the project not only aids in understanding current events but also enhances skills in data analysis, text processing, and Python programming. The resulting application serves as a comprehensive tool for journalism students and enthusiasts to explore trends, sentiments, and key information from the news.

Introduction

In today's fast-paced digital world, staying informed about current events and trends is more important than ever, especially for students and professionals in the fields of mass communication and journalism.

The News Aggregator and Analyzer project seeks to address this challenge by providing a comprehensive tool to collect, categorize, analyze, and visualize news articles from diverse sources. This Python-based application leverages various programming concepts and libraries to fetch news articles, categorize them into relevant topics, extract important keywords, and perform sentiment analysis. By presenting the analyzed data through intuitive visualizations, this project aims to simplify the process of understanding and interpreting the vast amounts of news data available.

Designed to enhance the skills and knowledge of students and enthusiasts in the field of journalism, this project not only offers a practical application of Python programming but also provides valuable insights into the world of news and information. Through this project, users will gain experience in web scraping, natural language processing, data analysis, and visualization, making it a valuable learning tool and a powerful resource for staying up-to-date with current events.

Project Detail:

Enhanced News Aggregator and Analyzer Project

1. **Importing Libraries**
2. **Fetching News Articles**
3. **Categorizing Articles**
4. **Keyword Extraction**
5. **Analyzing Sentiment**
6. **Visualizing Data**
7. **Trend Analysis**
8. **Summary Generation**
9. **Main Function**

1. Importing Libraries

python

```
import requests
from bs4 import BeautifulSoup
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
import matplotlib.pyplot as plt
from collections import Counter
from wordcloud import WordCloud

nltk.download('vader_lexicon')
```

2. Fetching News Articles

python

```
def fetch_articles(url):
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')
    articles = []

    for item in soup.find_all('h2'):
        title = item.text
        link = item.find('a')['href']
        articles.append((title, link))

    return articles
```

3. Categorizing Articles

python

```
def categorize_articles(articles):
    categories = {
        'politics': [],
        'sports': [],
        'entertainment': [],
        'technology': [],
    }

    for title, link in articles:
        if 'politics' in title.lower():
            categories['politics'].append((title, link))
        elif 'sports' in title.lower():
            categories['sports'].append((title, link))
        elif 'entertainment' in title.lower():
            categories['entertainment'].append((title, link))
        elif 'technology' in title.lower():
            categories['technology'].append((title, link))
        else:
            categories['others'].append((title, link))

    return categories
```

4. Keyword Extraction

python

```
def extract_keywords(articles):
    keywords = Counter()

    for title, link in articles:
        words = title.split()
        for word in words:
            keywords[word.lower()] += 1

    return keywords
```

5. Analyzing Sentiment

python

```
def analyze_sentiment(articles):
    sia = SentimentIntensityAnalyzer()
    sentiments = {}

    for title, link in articles:
        sentiment = sia.polarity_scores(title)['compound']
        sentiments[title] = sentiment

    return sentiments
```

6. Visualizing Data

python

```
def visualize_data(sentiments):
    titles = list(sentiments.keys())
    scores = list(sentiments.values())

    plt.figure(figsize=(10, 5))
    plt.barh(titles, scores, color='skyblue')
    plt.xlabel('Sentiment Score')
    plt.ylabel('Article Titles')
    plt.title('Sentiment Analysis of News Articles')
    plt.tight_layout()
    plt.show()
```

7. Trend Analysis

python

```
def trend_analysis(keywords):
    words = list(keywords.keys())
    counts = list(keywords.values())

    plt.figure(figsize=(10, 5))
    plt.plot(words, counts, marker='o')
    plt.xlabel('Keywords')
    plt.ylabel('Frequency')
    plt.title('Keyword Trend Analysis')
    plt.xticks(rotation=90)
    plt.tight_layout()
    plt.show()
```

8. Summary Generation

python

```
def generate_summary(articles):
    summaries = {}

    for title, link in articles:
        response = requests.get(link)
        soup = BeautifulSoup(response.text, 'html.parser')
        paragraphs = soup.find_all('p')
        summary = ' '.join([p.text for p in paragraphs[:3]])
        summaries[title] = summary

    return summaries
```

9. Main Function

python

```
def main():
    url = 'https://example-news-website.com'
    articles = fetch_articles(url)
    categories = categorize_articles(articles)
    keywords = extract_keywords(articles)
    sentiments = analyze_sentiment(articles)
    summaries = generate_summary(articles)

    visualize_data(sentiments)
    trend_analysis(keywords)

    print("Categories: ", categories)
    print("Keywords: ", keywords)
    print("Summaries: ", summaries)

if __name__ == '__main__':
    main()
```

Step 5: Running the Code

Run the `news_aggregator.py` file from your terminal or command prompt:

bash

```
python news_aggregator.py
```

Discussion

The **News Aggregator and Analyzer** project is an engaging and educational endeavor that combines multiple aspects of Python programming and data analysis. Here's a deeper dive into various components and the learning outcomes associated with each:

1. Web Scraping

Web scraping is a valuable skill for collecting data from websites. In this project, we use the `requests` library to fetch web pages and `BeautifulSoup` to parse and extract information. This helps in gathering a significant amount of news data for analysis.

Learning Outcomes:

- Understanding HTTP requests and responses.
- Learning to parse HTML and extract specific elements.
- Handling common issues such as missing elements or rate limiting.

2. Data Categorization

Categorizing articles into different topics is essential for organizing information. We use conditional statements and string operations to determine the category based on keywords in the title.

Learning Outcomes:

- Implementing basic string operations and conditional statements.
- Creating and managing dictionaries for categorization.
- Enhancing data organization and retrieval skills.

3. Natural Language Processing (NLP)

NLP techniques like sentiment analysis and keyword extraction are crucial for understanding the content and context of the news articles. Libraries such as `nltk` provide powerful tools for these tasks.

Learning Outcomes:

- Introduction to NLP and its applications.
- Performing sentiment analysis to gauge public opinion.
- Extracting keywords to identify important topics and trends.

4. Data Visualization

Visualizing data helps in understanding patterns and trends. Using libraries like `matplotlib` and `WordCloud`, we can create bar charts, line graphs, and word clouds to present our analysis visually.

Learning Outcomes:

- Creating various types of plots and charts.
- Customizing visualizations for better clarity and presentation.
- Using visual tools to communicate data insights effectively.

5. Trend Analysis

Analyzing keyword trends over time provides insights into changing topics of interest. This involves manipulating and plotting data to observe patterns.

Learning Outcomes:

- Using lists, sets, and tuples for data manipulation.
- Plotting and interpreting trends over time.
- Understanding the implications of changing trends.

6. Summarization

Generating summaries of news articles is a valuable skill for condensing information. This requires parsing article content and extracting key points.

Learning Outcomes:

- Extracting and summarizing text data.
- Implementing basic text processing techniques.
- Creating concise summaries that capture the essence of the content.

Conclusion

The News Aggregator and Analyzer project is a comprehensive and multi-faceted undertaking that provides a practical application of Python programming, data analysis, and natural language processing. It not only equips students with essential technical skills but also enhances their ability to interpret and analyze large volumes of information. By completing this project, students gain hands-on experience in various domains, making it an invaluable addition to their learning journey..

The News Aggregator and Analyzer project seeks to address this challenge by providing a comprehensive tool to collect, categorize, analyze, and visualize news articles from diverse sources. This Python-based application leverages various programming concepts and libraries to fetch news articles, categorize them into relevant topics, extract important keywords, and perform sentiment analysis. By presenting the analyzed data through intuitive visualizations, this project aims to simplify the process of understanding and interpreting the vast amounts of news data available.

Designed to enhance the skills and knowledge of students and enthusiasts in the field of journalism, this project not only offers a practical application of Python programming but also provides valuable insights into the world of news and information. Through this project, users will gain experience in web scraping, natural language processing, data analysis, and visualization, making it a valuable learning tool and a powerful resource for staying up-to-date with current events.