**Summer 2022 Programming Challenge Submission**

**OUR DATA –**

Web scraped Data from the site using the python library Beautiful Soup. <https://www.aljazeera.com/where/mozambique/>

Method-

1. Extracted the HTML content using the requests library.
2. Analyzed the HTML structure and identify the tags which have our content.
3. Extracted the tags using Beautiful Soup and put the data in a Python list.

Graphical user interface, application

Description automatically generated**JSON REPRESENTATION OF THE SCRAPED DATA in the Scraped\_Data.json file**

* It contains the index of the article ie 0-9
* Article title
* Article content
* Article link

**TQDM library usage for tracking the progress and calculating the time taken for the operation of the code-**



The time taken for the scraping and the analysis of the articles is 3 seconds.

**PLOTS USING PLOTLY-**

Chart, sunburst chart

Description automatically generated

Chart

Description automatically generated

Chart

Description automatically generated with medium confidence

**SENTIMENT ANALYSIS BY 3 METHODS**

**1)TEXTBLOB**

[Textblob](https://textblob.readthedocs.io/en/dev/) sentiment analyzer returns two properties for a given input sentence:

* **Polarity** is a float that lies between [-1,1], -1 indicates negative sentiment and +1 indicates positive sentiments.
* **Subjectivity** is also a float that lies in the range of [0,1]. Subjective sentences generally refer to opinion, emotion, or judgment.

**2)VADER Sentiment**

It uses a list of lexical features (e.g. words) which are labeled as positive or negative according to their semantic orientation to calculate the text sentiment.

Vader sentiment returns the probability of a given input sentence to be positive, negative, and neutral.

### ****3)FLAIR****

### [Flair](https://github.com/flairNLP/flair) is a simple to use framework for state-of-the-art NLP. It provided various functionalities such as:

### pre-trained sentiment analysis models,

### text embeddings,

### NER

**OVERVIEW OF MY CODE**

* After web scraping and extracting the articles from the link I combined the different paras obtained from one article.
* Performed some text cleaning to get rid of escape characters, spaces, and stopwords.
* Iterated over the sentences of the article to predict the sentiment of each using the sentiment metric and stored it in the form of a list.
* The polarity\_data and subjectivity\_data are filled respectively with the values returned by the metric for analysis.
* I calculated the average of the polarity\_data and the subjectivity\_data of all the sentences of the article and returned the overall sentiment of the article using the function find\_sentiment.

Text

Description automatically generated

* In the find\_sentiment function I have declared threshold values of 9 categories of sentiments to get a better understanding of the article in literal ways. The average polarity score is mapped to the sentiments declared.
* For calculating the polarity and the subjectivity score I have taken into account 40% of the article title analyzed score and 60% of the article content analyzed score

Text

Description automatically generated

* I have created dataframes using the pandas library for visualizing the results obtained.
* The article\_df dataframe has Article title, Article content, and the Article link as the attributes.

Graphical user interface, application

Description automatically generated

* Conversion and reading the data to and from these dataframes into any JSON, CSV format is coded as well.
* The SentimentIntensityAnalyzer of the VaderSentiment package reads the dataframe and not only tells about the Polarity score, in addition, it tells us how positive or negative a conclusion is. I have computed negative, neutral and positive scores returned for each article giving us deeper insights of the article sentiment.
* Since most of the articles had a negative notion according to the values returned by FLAIR, I tried to analyze the negative computed values by the VaderSentiment and plotted a Pie chart representing the % of the negative notion of these 10 articles in comparison to each other.
* The prediction of the sentiment by the FLAIR library was mostly biased towards the negative sentiment. This I noticed while comparing the results obtained by the three methods.

**COMPARISON OF ALL THE THREE METHODS-**

VADER not only tells us about the Polarity score, in addition, it tells us about how positive or negative a conclusion is. The Neutral, Negative, and Positive scores help in the analysis of how neutral or how negative the article is.

Text

Description automatically generated with low confidence

The flair sentiment classifier produced results different from that of the Textblob and VaderSentiment. The flair sentiment classifier was originally trained on IMDB movie review data and not Hacker News comments. This difference may explain some of the limitations of this classifier, but it also appears that the TextBlob library captures other dimensions of “sentiment” that the flair NLP library just doesn’t have.

**HOW TO RUN THE CODE-**

The commands in the .ipynb notebook can be run as the whole file or individual cells as well.

**WORDCLOUD of the 10 articles scraped from the site**-

Generated an image using wordcloud that gives us the most representative words in the 10 scraped articles

Text

Description automatically generated with medium confidence