

Name: Urulugastenne Amarakone

Student Reference Number: 10899177

Module Code: PUSL3189	Module Name: Natural Language Processing
Coursework Title: Stock Market Based Sentiment Analysis	
Deadline Date: 2/01/2025	Member of staff responsible for coursework: Miss. Nethmi Weerasinghe
Programme: BSc (Hons) Data Science	
Please note that University Academic Regulations are available under Rules and Regulations on the University website www.plymouth.ac.uk/studenthandbook .	
<p>Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.</p> <p><i>We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.</i></p> <p>Signed on behalf of the group:</p>	
<p>Individual assignment: <i>I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.</i></p> <p>Signed: Ruchira</p>	
<p>Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.</p> <p>I *have used/not used translation software.</p> <p>If used, please state name of software.....</p>	
<p>Overall mark _____ % Assessors Initials _____ Date _____</p>	

Table of Contents

Introduction	5
Data collection.....	6
Objective	6
Methodology:	6
Code Snippet:	7
Dependencies	7
Text Preprocessing	7
Objective	7
Stop word Removal:	7
Code Snippet:	8
Lemmatization:	8
Code Snippet:	9
Tokenization:	9
Code Snippet:	9
N-Grams:.....	9
Code Snippet:	10
Special Character Removal:.....	10
Code Snippet:	10
Dependencies:	11
Part-of-Speech Tagging and Named Entity Recognition	11
Objective:	11
Methods:	11
Part-of-Speech (POS) Tagging:	11
Named Entity Recognition (NER):	11
Code Snippet:	12
.....	12
Dependencies:	12
Sentiment Analysis	12
Objective	12
Methods:	12
Overview:.....	12
Stock Market Relevance:	13
Technologies and Libraries Used.....	13

Implementation:	13
Output:	13
Code Snippet:	14
Generating plots:	14
Word cloud:	15
Code Snippet:	17
Dependencies:	17
Topic Modeling	17
Objective:	17
Importance:	17
Methods:	17
Code Snippet:	18
Dependencies:	18
Extracted Topics and Interpretation:	18
Stylometric Analysis and Visualization	19
Objective:	19
Methodology:	19
Dimensionality Reduction:	19
Code Snippet:	20
Clustering:	21
Code Snippet:	22
Dendrogram Visualization (Hierarchical Clustering)	22
Code Snippet:	23
Visualization:	23
Dependencies:	23
Document Clustering with Word2Vec or Doc2Vec	24
Objective:	24
Methodology:	24
Code Snippet:	24
Visualization:	24
PCA	24
Code Snippet:	26
t-SNE	27
Interpretation of Results:	28
Dependencies:	28
Code Snippet:	28

Dependency Parsing and Advanced Structures28

Key Concepts: 28

Implementation Steps:..... 29

Code Snippet: 30

Dependencies: 30

Insights and Real-World Applications 30

Objective: 30

Key Insights: 30

Impact on Decision-Making: 31

Summary: 32

Implementing Advanced NLP Techniques for text summarization.....32

Objective: 32

Methodology: 32

Code Snippet: 33

Results and Significance: 34

Expected Outcomes: 35

Conclusion35

References36

1. Introduction

As a subfield of artificial intelligence Natural language processing (NLP) is a field that focuses on the interaction between machine language and human language. NLP is a popular technology that is used to stock market prediction, social media analysis and business intelligence to extract meaningful insights from unstructured text data. This data can be manipulated using NLP while removing their unnecessary data and structure them in a correct order that data can be used to feed numerous techniques that generate insights from text data financially beneficial.

In a stock market buyer's sentiment plays a major role that can directly affect the direction of the price movement of a stock while some stock markets like Colombo Stock exchange, Bombay stock exchange are completely lying on the news sentiment. Positive news can create positive sentiment on the market and can convert the market direction to a bull trend (uptrend). Also, negative news can convert the direction of the market to a down trend which is commonly called a bear market. News about economic growth, corporate earnings, political stability or business friendly policy changes are considered as positive sentiment and political instability, economic red flags or adverse regulations lead to negative sentiment.

Each day investors need to get a clear understanding about market sentiment by going through all the news before buying or selling a stock. Reading all the news articles and government policy changes everyday morning before the action call of a market is a tough task because there is so many unnecessary news available in news web sites but need to go through all news articles searching for valuable information.

The aim of this project is to extract data from Sri Lankan business news websites and analyze them and categorize them as positive, negative or neutral and generate a decision about overall sentiment of the market. Also, this project helps to reduce time and effort investors need to put in news article reading. The analysis focuses on business news from prominent Sri Lankan business news web sites, including

- . Ada Derana BIZ
- Daily Mirror Business
- News webpage of Central Bank of Sri Lanka
- Economy times Global Business news

By observing the data from these reliable web sources. This project seeks to predict the sentiment climate of Colombo stock market.

2. Data collection

Objective

Collect text data related to business and finance category from reliable sources to provide a diverse corpus for the analysis, also that can impact on Colombo Stock Exchange.

Methodology:

- Step 1: Identifying Reliable News Sources
 - For the data extraction process reputable and reliable news sources are compulsory to make dictions. Web sites such as Central Bank of Sri Lanka, Ada Derana, Daily Mirror, and global news platforms like Economy Times India Business, Economy Times India Stock news is chosen to ensure data relevance, reliability and dynamic updates.
- Step 2: Web Scraping Implementation
 - To extract textual content such as headlines, article bodies and their publication dates from news web sites **BeautifulSoup** library is used to scrape HTML web pages.
- Step 3: Filtering Relevant Content
 - As the data filtering process, unnecessary text is removed using a threshold value. Also, the texts that contain small words count because that kind of news cannot generate a proper meaning. After this filtering process all the data is loaded to “df” data frame.
- Step 4: Handling Data Quality
 - Duplicated news articles can generate noise and overfitting. Duplicated data is removed by comparing their textual content and using hashing techniques to establish the data quality of the data set.
- Step 5: Structuring the Data
 - Extracted data is loaded in to separate four data frames and combined them together into a one data frame that helps to remove duplicate news articles and the article that contains same news. Each row of the data frame contains different news articles.

Code Snippet:

extracting data from ada derana biz

```
[ ]
```

```
# Fetch the webpage content
url = "https://bizenglish.adaderana.lk/"
response = requests.get(url)

# Parse the content using BeautifulSoup
soup = BeautifulSoup(response.content, "html.parser")

# Find all <p> tags and clean the extracted text
p_texts = [p.get_text().strip() for p in soup.find_all('p') if p.get_text().strip()]

# Load the cleaned data into a pandas DataFrame
df = pd.DataFrame(p_texts, columns=["Paragraph Text"])

# Display the cleaned DataFrame
print(df.head())
```

Figure 1: Text extraction from web

Dependencies

- requests: To send HTTP requests to fetch web pages
- BeautifulSoup: To phrase and extract the content from html

3. Text Preprocessing

Objective

Clean and preprocess the text content before the analysis.

Steps

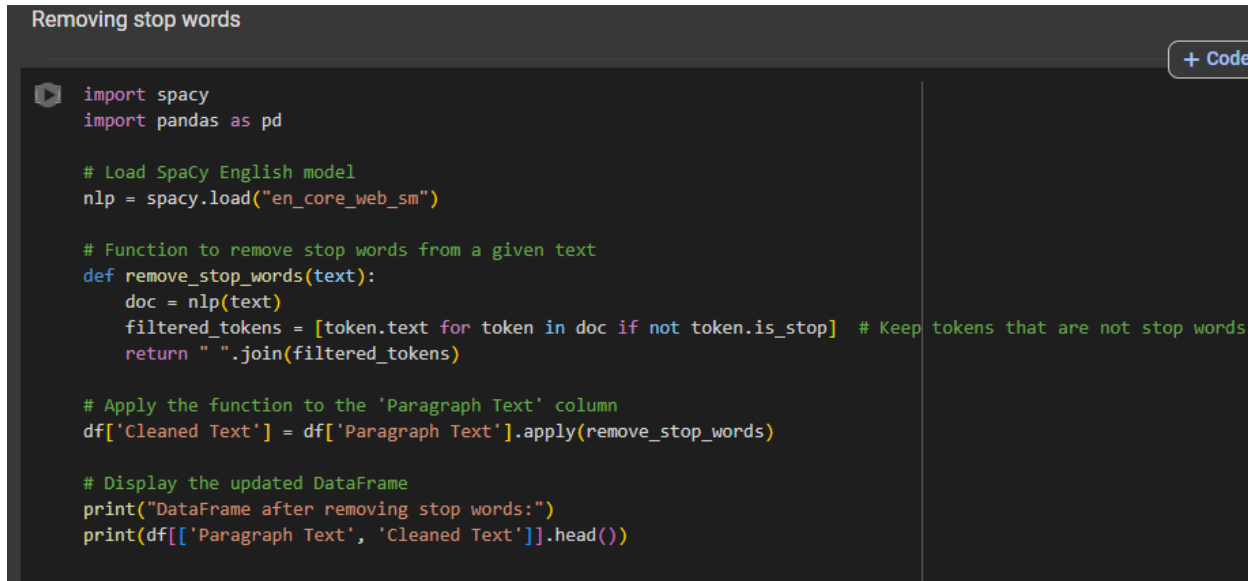
Stop word Removal:

- Stopwords are words that commonly appear in text contents. Words such as “the”, “and”, “as” are some common examples for stop words. When considering larger text data sets the

amount of stop words is large and that does not contribute significantly to the semantic meaning also create noise in text data set. The noise of stop-words can lead to false insights.

- Spacy library is used to remove the text data from “**pharagraph text**” column from “**df** ” and stored in a new column called “Cleaned text” data frame .Spays build in “**en_core_web_sm**” model is utilized to remove all the stop words.

Code Snippet:



```
import spacy
import pandas as pd

# Load SpaCy English model
nlp = spacy.load("en_core_web_sm")

# Function to remove stop words from a given text
def remove_stop_words(text):
    doc = nlp(text)
    filtered_tokens = [token.text for token in doc if not token.is_stop] # Keep tokens that are not stop words
    return " ".join(filtered_tokens)

# Apply the function to the 'Paragraph Text' column
df['Cleaned Text'] = df['Paragraph Text'].apply(remove_stop_words)

# Display the updated DataFrame
print("DataFrame after removing stop words:")
print(df[['Paragraph Text', 'Cleaned Text']].head())
```

Figure 2: Removing stop words

Lemmatization:

- Lemmatization is concept that convert words to reduces their root form without changing the meaning of the word. (e.g., "going" becomes "go"). Lemmatization is highly beneficial when visualizing most common words, also for the sentiment analyzing.
- Data from “**Cleaned text** “column in “**df**” data frame are lemmatized using Spacy library’s build in model’s pipeline. Lemmatized words are organized in “**Tokens_from_stop**” column to improve the performance of downstream tasks.

Code Snippet:

```
lemmatization

[ ] import spacy
import pandas as pd

# Load SpaCy English model
nlp = spacy.load("en_core_web_sm")

# Function to perform lemmatization
def lemmatize_text(text):
    doc = nlp(text)
    lemmatized_tokens = [token.lemma_ for token in doc if not token.is_stop and not token.is_punct]
    return " ".join(lemmatized_tokens)

# Assuming df is your DataFrame with a 'Paragraph Text' column
df['Lemmatized Text'] = df['Paragraph Text'].apply(lemmatize_text)

# Display the updated DataFrame
print("DataFrame after Lemmatization:")
print(df[['Paragraph Text', 'Lemmatized Text']].head())
```

Figure 3:lemmatization

Tokenization:

- Breaking down text content into smaller pieces called word tokenization. Involvement of tokenization is essential for granular level text analysis.
- The spacy library's build in English language pipeline is employed to preform text tokenization on “**Cleaned text**” column in” df” data frame. And all the tokenized words are loaded into a new column called “**Tokens_from_stop**” to feed downstream tasks.

Code Snippet:

```
lemmatization

[ ] import spacy
import pandas as pd

# Load SpaCy English model
nlp = spacy.load("en_core_web_sm")

# Function to perform lemmatization
def lemmatize_text(text):
    doc = nlp(text)
    lemmatized_tokens = [token.lemma_ for token in doc if not token.is_stop and not token.is_punct]
    return " ".join(lemmatized_tokens)

# Assuming df is your DataFrame with a 'Paragraph Text' column
df['Lemmatized Text'] = df['Paragraph Text'].apply(lemmatize_text)

# Display the updated DataFrame
print("DataFrame after Lemmatization:")
print(df[['Paragraph Text', 'Lemmatized Text']].head())
```

Figure 4:tokenization

N-Grams:

- **Context Capture:** N-grams represent word sequences, capturing relationships and context in text.

- **Enhanced Features:** They enrich text data with meaningful patterns for classification, prediction, and sentiment analysis.

Code Snippet:

Creating n grams

```
[ ] from nltk.util import ngrams

# Function to create n-grams
def create_ngrams(text, n):
    tokens = text.split() # Split the text into words (assuming space-separated tokens)
    n_grams = list(ngrams(tokens, n)) # Generate n-grams
    return [" ".join(gram) for gram in n_grams]

# Create n-grams (example: bigrams and trigrams)
df['Bigrams'] = df['Cleaned Text'].apply(lambda text: create_ngrams(text, 2)) # n=2 for bigrams
df['Trigrams'] = df['Cleaned Text'].apply(lambda text: create_ngrams(text, 3)) # n=3 for trigrams

# Display the updated DataFrame
print("DataFrame with N-grams:")
print(df[['Paragraph Text', 'Bigrams', 'Trigrams']].head())
```

Figure 5:N-gram creation

Special Character Removal:

- Symbolic marks, emojis and Punctuation marks available in the text can introduce noise into the text data set and that may exist for issues of text processing.
- Expressions from “REGEX” library were used to strip out those characters from text content before Stop words, lemmatization or tokenization task preform.

Code Snippet:

Removing unnecessary marks using regex

```
[ ] # Example: Assuming df has a column named 'Paragraph'
# Regex to remove unnecessary marks (punctuation, special characters, etc.)
df['Paragraph Text'] = df['Paragraph Text'].str.replace(r'^\w\s', '', regex=True)

# Explanation:
# [^\w\s] - Matches anything that is NOT a word character (\w) or whitespace (\s)
# '', replaces matches with an empty string.

# Display the cleaned DataFrame
print("Cleaned Paragraph Column:")
print(df[['Paragraph Text']].head())
```

Figure 6:Removing special characters

Dependencies:

- spacy: For tokenization, lemmatization and removing stop words.
- re: To remove symbols, punctuation marks and clean the text content.

4. Part-of-Speech Tagging and Named Entity Recognition

Objective:

Part of the speech and Named Entity Recognition is used to extract significant entities from the text and label them.

Methods:**Part-of-Speech (POS) Tagging:**

- Assigning grammatical labels to divided tokens from text called POS tagging. Nouns, verbs and adjectives are commonly available in every text content Pos is utilized to understand their positioning order in a sentence and helps to identify their roles and relationships.
- Spacy library's "**en_core_web_sm**" model is employed to for this Pos tagging task. Which provides accurate POS tagging capabilities. Data from "**df**" data frames "**Cleaned text**" column is converted to Pos tags and stored them in a separate column called "**POS Tags**".
- As an example, after feeding this sentence "Forex market is crashed," to the pipeline POS tagging identifies "Forex" as a noun, "market" as a noun, and "crashed " as a verb.

Named Entity Recognition (NER):

- Named Entity Recognition (NER) is used to identify and named entities available in the text contents. Institution/organization names, people, locations and dates are identified as entities by NER algorithm.
- Correctly identifying organization names is the objective of this project. Extracting NER from news articles are done by utilizing Spacy library's "**en_core_web-sm**" model. This model is accurate in identifying named entities. Correctly recognized dates and company names are added in to a new column named "**NER**" to use in downstream tasks.
- This step of Named Entity Recognition is crucial to correctly identifying news sentiment and investor behavior in the stock market.

Code Snippet:

Pos and Ner

```
[ ]

# Load SpaCy English model
nlp = spacy.load("en_core_web_sm")

# Function to perform POS tagging and Named Entity Recognition (NER)
def pos_and_ner(text):
    doc = nlp(text) # Process the text with SpaCy
    pos_tags = [(token.text, token.pos_) for token in doc] # POS tagging
    named_entities = [(ent.text, ent.label_) for ent in doc.ents] # Named entities
    return pos_tags, named_entities

# Assuming your DataFrame has 'Paragraph Text' and 'Tokens'
# Apply POS tagging and NER on 'Paragraph Text'
df['POS Tags'], df['Named Entities'] = zip(*df['Cleaned Text'].apply(pos_and_ner))

# Display the updated DataFrame
print("POS Tagging and NER completed. Sample output:")
print(df[['Paragraph Text', 'Tokens', 'POS Tags', 'Named Entities']].head())
```

Figure 7:Pos and Ner

.

Dependencies:

- Library: spacy.
- Mode: en_core_web_sm

5. Sentiment Analysis

Objective

Divide news articles into text data according to the sentiment as positive, negative. Especially news related to companies.

Methods:

Overview:

- To reveal customer sentiment of a stock market requires correctly identified news according to their emotional tone as positive, negative or neutral. The Sentiment of investors lead to market trends, investor trends and market liquidity. Sentiment analysis can predict the future direction of a market.

Stock Market Relevance:

- News articles about strong earnings reports, interest rate cuts and business friendly policy changes from government are creating positive sentiments on markets and they provide green signals about bullish market.
- Negative sentiment can be caused by increased interest rates, economic breakdowns and they are directly correlated to bearish market trends.

Technologies and Libraries Used:

- **VADER** (Valence Aware Dictionary and Sentiment Reasoner): Is a rule-based model that categorizes text content as positive, negative or neutral while providing scores for each sentiment type and identify overall sentiment with a compound score.
- **TextBlob**: TextBlob library offered an analysis that contains polarity and subjectivity. This used to complement VADERs process.
- **SentimentIntensityAnalyzer (SIA)**: To fine grained the sentiment scoring NLTK sentiment analyzer is employed.
- Python packages: nltk, vaderSentiment, textblob, spacy.

Implementation:

- Cleaned and preprocessed text data served as input for sentiment analysis models.
- SIA and VADER applied for granular sentiment scoring, particularly for headlines and article bodies.
- TextBlob's polarity scores added context, identifying nuances in overall sentiment trends.
- Sentiment-specific outputs included compound scores and sentiment categories (Positive, Negative, Neutral).

Output:

- Distribution of sentiment scores was loaded into two separate columns in “df” data frame those columns contain sentiment score and sentiment labels as two separate values.

Code Snippet:

Sentiment analysing

```
[ ]
from nltk.sentiment import SentimentIntensityAnalyzer

# Load SpaCy English model
nlp = spacy.load("en_core_web_sm")

# Load VADER for sentiment analysis
nltk.download('vader_lexicon')
sia = SentimentIntensityAnalyzer()

# Function to highlight institutions (ORG) and actions (VERB)
def highlight_institutions_and_actions(text, named_entities, pos_tags):
    highlighted_text = text

    # Highlight institutions (ORG) from Named Entities
    institutions = [ent for ent, label in named_entities if label == 'ORG']
    for institution in institutions:
        highlighted_text = highlighted_text.replace(institution, f"\033[1;34m{institution.upper()}\033[0m")

    # Highlight actions (VERB) from POS Tags
    actions = [word for word, pos in pos_tags if pos == 'VERB']
    for action in actions:
        highlighted_text = highlighted_text.replace(action, f"\033[1;32m{action.upper()}\033[0m")

    return highlighted_text

# Function to determine sentiment category
def categorize_sentiment(text):
    sentiment_score = sia.polarity_scores(text)['compound']
    if sentiment_score > 0.05:
        return 'Positive', sentiment_score
    elif sentiment_score < -0.05:
        return 'Negative', sentiment_score
    else:
```

Figure 8:sentiment analysis and plot generating (code is too lengthy to capture the full code)

Generating plots:

- As the data visualization step of sentiment analysis “sentiment” column is used to calculate overall counts from three sentiment tone and. Using “Matplotlib” and “seaborn” library’s the graph below was created.

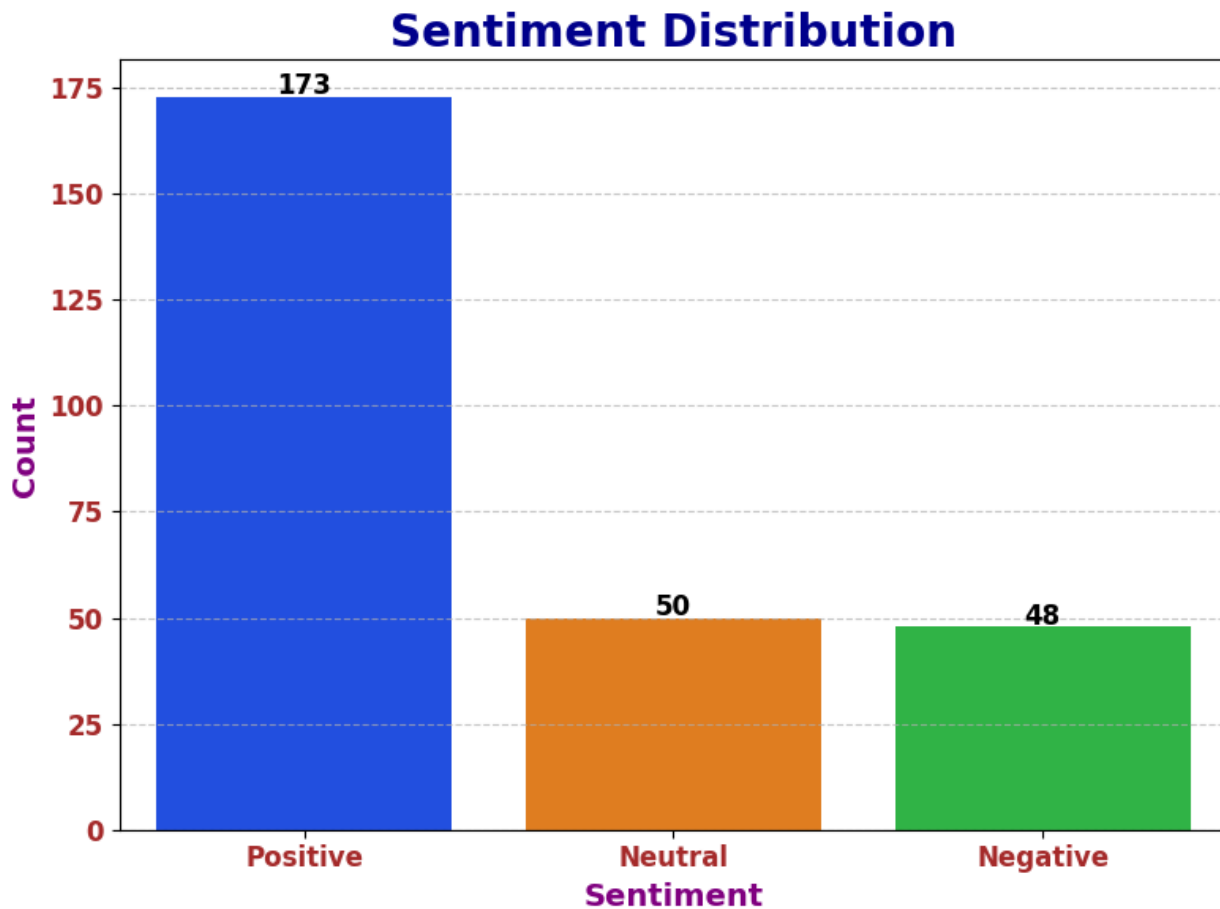


Figure 9:Sentiment Visualization Bar plot

This bar chart contains the overall news sentiment of Sri Lankan Stock market including international news that can affect on financial markets. The highest number of news articles are generating positive sentiment while securing a significant gap from negative and neutral news. The number of negative news and neutral news are extremely like each other. Also, this analysis shows green flags about an upcoming uptrend. (This project is based on real time data from news websites and these values might be changed based on time and dates.

Word cloud:

- Word cloud is a data visualization method that is used to visualize the frequency of words in a data set. Most common words are represented in larger and bolder.

[illegible]

Figure 10: Word Cloud

According to the word cloud “Sri Lanka” is the most prominent word. That indicating policy changes or business decisions related to Sri Lanka. Significant decisions of monetary policies are highlighted by “Central Bank “the second most appeared word. Regular references to economic metrics like interest rates or other financial indicators are highlighted by words such as “percent” and “rate”. Discussions around business institutions are printout by “company” and sector words”. Market, Development and stock world provides green signals about market trends and market performance. Overall sentiment of the word cloud provides signals about positive news articles.

Code Snippet:

```
Word cloud

[ ] from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Flatten the lists in 'Tokens_from_stop' column and combine into a single string
text_data = " ".join([" ".join(tokens) if isinstance(tokens, list) else tokens for tokens in df['Tokens_from_stop']])

# Generate the word cloud
wordcloud = WordCloud(
    width=800,
    height=400,
    background_color='white',
    colormap='viridis',
    max_words=200
).generate(text_data)

# Plot the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Word Cloud from 'Tokens_from_stop'", fontsize=16, fontweight='bold')
plt.show()
```

Figure 11:creating Word Cloud

Dependencies:

- Python libraries: nltk, vaderSentiment, textblob, spacy, matplotlib, pandas.

6. Topic Modeling

Objective:

To understand recurring trends and themes, correctly identify the keys in the collected text data set.

Importance:

This topic modeling concept provides a structured overview of the data by identifying hidden patterns in a large text corpus. Topic modeling is an efficient pathway for discovering hidden patterns within large text contents. Related topics are grouped according to their distinct themes. By highlighting major topics, topic modeling helps to simplify the data set and provides insights for decision making.

Methods:

1. Latent Dirichlet Allocation (LDA):

- Using probability concepts, the LDA model can identify recurring topics in the text corpus.
- Each topic inside the text corpus is represented by a distribution of words. Also, in every text document, they were treated as a mixture of multiple topics.

2. Keyword Extraction:

- The process of keyword extraction is critical to understand the primary focus using extracted representative.

- As an example, topics related to "economic growth" are directly correlated to phrases like "GDP" (gross domestic product) "inflation," and "exports."

3. Major Topic Extraction:

- To extract five major topics from the text, the "Gensim" library is employed to perform **Latent Dirichlet Allocation (LDA)** for topic modeling tasks. Using text corpus data represented as a bag of words dictionary, it performs its task via training an LDA model. By observing the list of top keywords and their weights, the top five topics are identified by the model. Key parameters to print out the major topics are assigned into `num_topics` (number of topics) variable, `passes` (training iterations) variable are used to setup the commands about the necessary iterations need to train. The topics and their keywords are printed to summarize and highlight the main themes inside the text data set as the final step. Coherence score of LDA model is 0.32 that indicates topics are not very interpretable.

Code Snippet:

```
[ ] # Number of topics to extract
    num_topics = 5

    # Train the LDA model
    lda_model = models.LdaModel(
        corpus=corpus,
        id2word=dictionary,
        num_topics=num_topics,
        random_state=42,
        passes=10,
        per_word_topics=True
    )

    # Display the topics and keywords
    print("Extracted Topics:")
    topics = lda_model.print_topics(num_words=10)
    for topic_num, topic_keywords in topics:
        print(f"Topic {topic_num + 1}: {topic_keywords}")
```

Figure 12: Extracting five major topics

Dependencies:

- Python libraries: gensim, pyLDAvis, matplotlib.

Extracted Topics and Interpretation:

- **Topic 1:** By highlighting key words such as "price," "trade" and "bank", first major topic is discussed about policy changes, influenced market trends and recent activity inside banks etc. As a summary, the first topic is focused on Sri Lanka's current economic environment.
- **Topic 2:** Signs for market trend, upward momentum of the markets are discussed by keywords like "increase", "present" and "market". Information about bullish market trends and company performance is interpreted by topic two. Also, this acts as an economic growth indicator.

- **Topic 3:** Highlighted key points such as “index,” “manufacturing” and “central” reviles about microeconomic updates of Sri Lanka.
- **Topic 4:** Insights related to stock market updates emphasized by terms like "stock," "year," and "percent." Also, it focuses on market analytics and stock performance.
- **Topic 5:** Topic five is discussing monetary policies with key points such as "policy," "rate," and "monetary." Generated analysis from this topic is mainly centered to decisions related to Central Bank of Sri Lanka.

7. Stylometric Analysis and Visualization

Objective:

Utilizing data visualization concepts and observing writing patterns of news articles. Preform **Stylometric Analysis and Visualization** to understand patterns, characteristics unique to writers and clustering similar articles from text data.

Methodology:

1. Feature Extraction:

- Length of sentences, word length, usage of punctuation marks and vocabulary richness (stylistic features) measured to perform downstream tasks.
- Calculate lexical diversity through -Token Ratio (TTR).
- For the step of quantifying the significance of words inside individual news articles that relative to the whole dataset **TF-IDF Vectorization** was applied. Also, TF-IDF can classify unique phrases.

Dimensionality Reduction:

- High-dimensional stylistic features are converted to two- or three-dimensional features for visualization using **Principal Component Analysis (PCA)** concept.

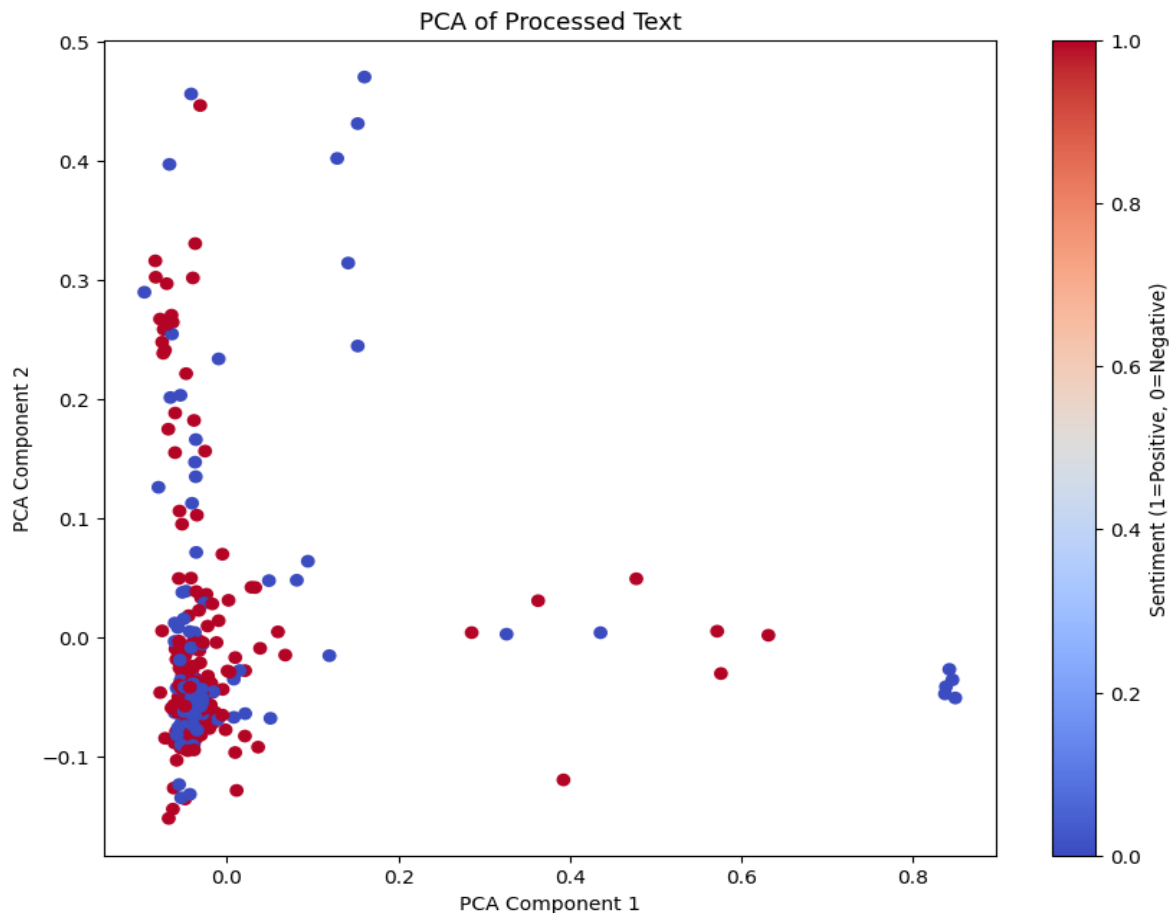


Figure 13:PCA plot

The above scatter plot was plotted to visualize the results Principal Component Analysis (PCA). The x-axis of the scatterplot labeled as "PCA Component 1," while capturing the most variance. PCA Component2 is represented by y axis of the scatterplot, and it captures the second most variance. Two color tones are used to represent two emotions of news articles. Blue colored points are used to represent negative emotions while red is used to showcase positive sentiment. A heatmap was generated to measure the strength of emotions. There is some clustering that suggests patterns in sentiment or features. Distribution and separability of sentiment classes is highlighted by this visualization plot.

Code Snippet:

PCA for Dimensionality Reduction

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

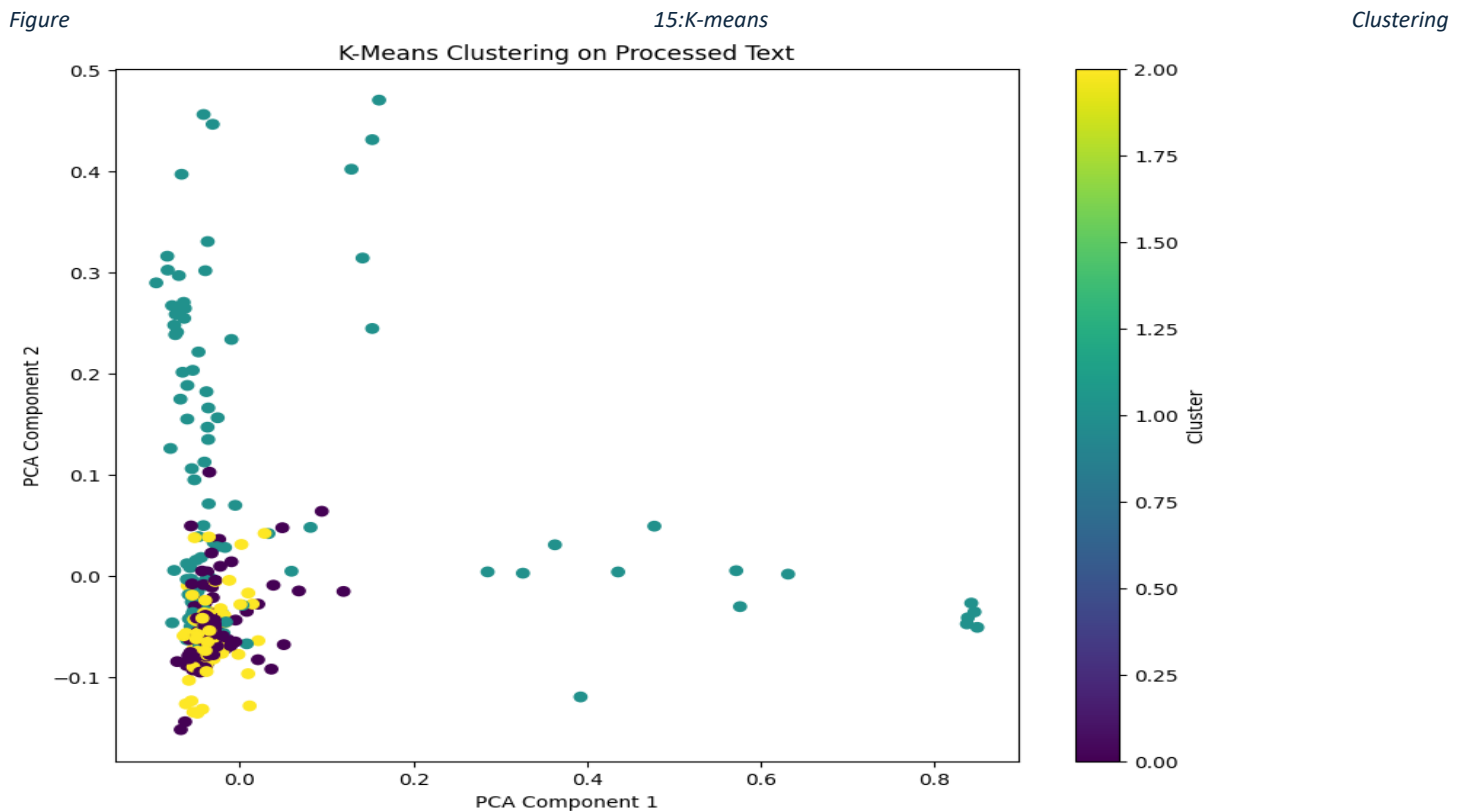
# Apply PCA for dimensionality reduction
pca = PCA(n_components=2)
pca_result = pca.fit_transform(tfidf_df)

# Plot the PCA results
plt.figure(figsize=(10, 8))
plt.scatter(pca_result[:, 0], pca_result[:, 1], c=df['Sentiment'].apply(lambda x: 1 if x == 'Positive' else 0), cmap='coolwarm')
plt.title('PCA of Processed Text')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.colorbar(label='Sentiment (1=Positive, 0=Negative)')
plt.show()
```

Figure 14:Principal Component Analysis

Clustering:

- News articles with similar stylistic patterns are grouped by K-Means clustering.
- Related insights into the average characteristics of a group are provided by Cluster centers



This scatter plot is used to visualize how K-Means clustering applied on processed text data with reduced dimensions after the PCA. X-axis of the cluster plot is labeled as “PCA component” and it contain the most variance in the data. PCA component 2 is the cluster that has second most variance according to this plot.

The x-axis represents "PCA Component 1," which captures the most variance in the data, while the y-axis shows "PCA Component 2," capturing the second most variance.

Data points are color-coded based on their cluster assignment, with the color bar on the right representing cluster labels (e.g., 0, 1, 2). Distinct colors indicate different clusters identified by the K-Means algorithm. Some clusters overlap or are tightly grouped, while others are more dispersed. This plot helps in understanding the clustering patterns of the text data in a two-dimensional space.

Code Snippet:

K-Means Clustering

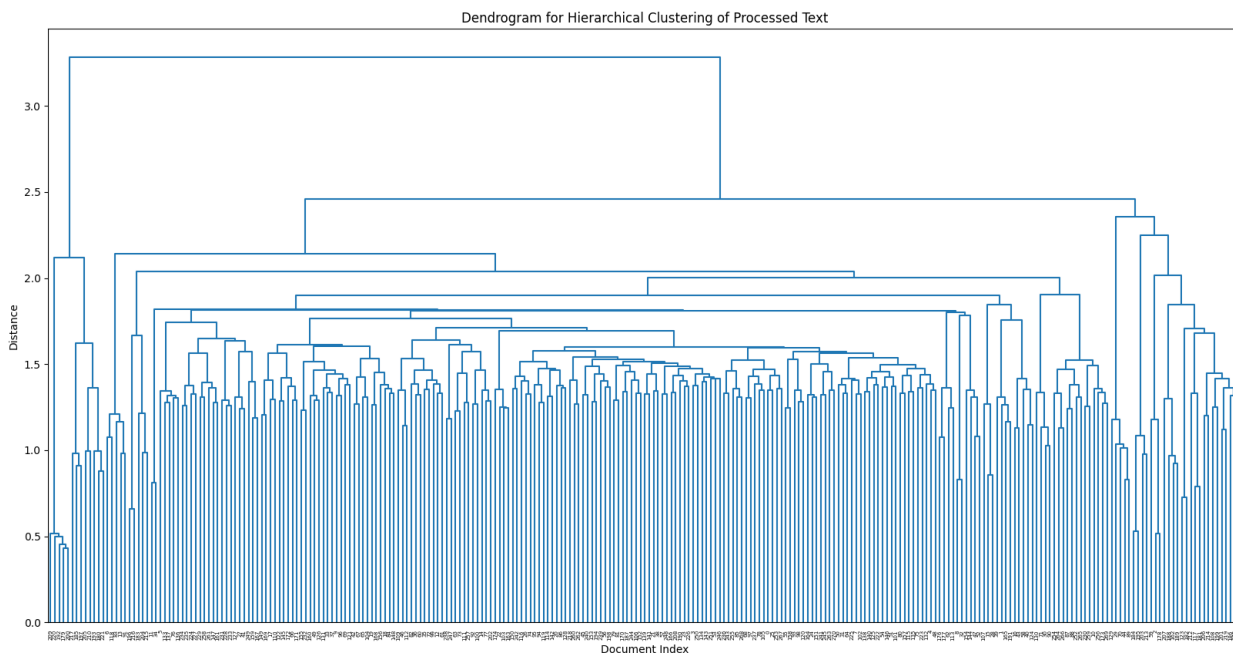
```
from sklearn.cluster import KMeans

# Apply KMeans clustering
kmeans = KMeans(n_clusters=3, random_state=42) # You can adjust the number of clusters
df['Cluster'] = kmeans.fit_predict(tfidf_df)

# Plot the PCA result with cluster labels
plt.figure(figsize=(10, 8))
plt.scatter(pca_result[:, 0], pca_result[:, 1], c=df['Cluster'], cmap='viridis')
plt.title('K-Means Clustering on Processed Text')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.colorbar(label='Cluster')
plt.show()
```

Figure 16:K-means Clustering

Dendrogram Visualization (Hierarchical Clustering)



This dendrogram provides a visualization of hierarchical relationships in between clusters of processed text. While observing this dendrogram it become easy to identify similarities of grouped documents based on their extracted features .The level of similarity is highlighted by its branches. Shorter branches represent closer relationships while longer branches indicate distant relationships. The goal of this visualization is to understand the structure of data. While revealing natural divisions that can highlight subsequent analysis or applications like categorization or targeted sentiment analysis.

Code Snippet:

Dendrogram Visualization (Hierarchical Clustering)

```
from scipy.cluster.hierarchy import dendrogram, linkage

# Perform hierarchical clustering
Z = linkage(tfidf_df, method='ward')

# Create a dendrogram
plt.figure(figsize=(20, 10))
dendrogram(Z, labels=df.index, orientation='top', color_threshold=0)
plt.title('Dendrogram for Hierarchical Clustering of Processed Text')
plt.xlabel('Document Index')
plt.ylabel('Distance')
plt.show()
```

Figure 17: Dendrogram Visualization

Visualization:

- Clusters are plotted using scatterplots and heatmaps plotted aligning with them for intuitive observations and insight extraction.
- Highlighted outliers and distinctive writing represented in specific clusters.

Dependencies:

- Python libraries: pandas, numpy, sklearn, matplotlib, seaborn.

8. Document Clustering with Word2Vec or Doc2Vec

Objective:

To group similar documents or sentences documents are clustered by advanced document representation techniques.

Methodology:

1. Document Vectorization:

- Via word2vec and Doc2vec techniques text data is converted into vector representations.
- For fine tuning parts Word2Vec/Doc2vec models are trained on the data set and leveraged the pre-trained embeddings.

2. Clustering Technique:

- To group similar documents or sentences K-Means clustering was applied on document vectors.
- Elbow method is used to determine the optimal number of clusters.

Code Snippet:

```
[ ] from gensim.models import Doc2Vec

model = Doc2Vec(vector_size=100, window=2, min_count=1, workers=4, epochs=20)
model.build_vocab(corpus)
model.train(corpus, total_examples=model.corpus_count, epochs=model.epochs)
```

Figure 18: Word2vec

Visualization:

- PCA and t-SNE are used to reduce dimensionality and visualize clusters.
- Scatter plots are generated to highlight similarities and cluster separation.

PCA

- Principal Component Analysis (PCA) converts data into a set of orthogonal components to reduce linear dimensionality. While explaining the maximum variance. And the algorithm provides yet informative and simplified data for visualization and clustering.

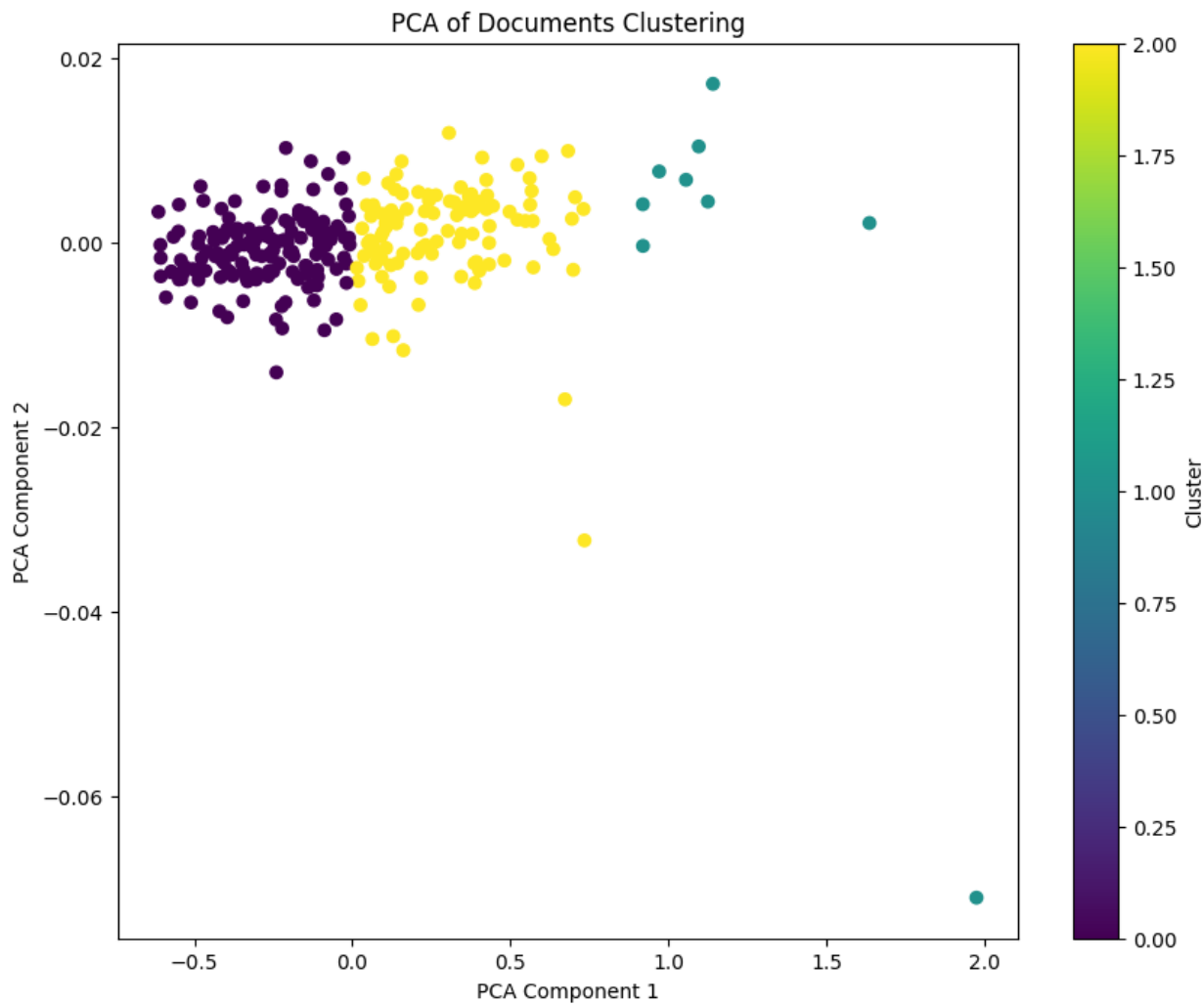


Figure 19:PCA plot W2V

The above scatterplot which titled its figure named "PCA of Documents Clustering" illustrate the document clustering results from PCA algorithm .A document is represented by each point. And the possession of the point is determined by the first two principal components. By observing the color of the point, we can identify which cluster it belongs to. This cluster plot highlights distinct groupings. Thematic similarities and differences among documents are highlighted by this scatterplot.

Code Snippet:

Visualize the Clusters Using PCA

```
[ ] from sklearn.decomposition import PCA
    import matplotlib.pyplot as plt

    # Apply PCA for dimensionality reduction
    pca = PCA(n_components=2)
    pca_result = pca.fit_transform(X)

    # Plot the PCA results
    plt.figure(figsize=(10, 8))
    plt.scatter(pca_result[:, 0], pca_result[:, 1], c=df['Cluster'], cmap='viridis')
    plt.title('PCA of Documents Clustering')
    plt.xlabel('PCA Component 1')
    plt.ylabel('PCA Component 2')
    plt.colorbar(label='Cluster')
    plt.show()
```

Figure 20:PCA cluster plot2

t-SNE

- t-SNE is a technique used to reduce non-linear dimensionality and plot high dimensional data in a 2D, or 3D space. This is a most suitable method to identify tightly knit clusters because it preserves the local structures.

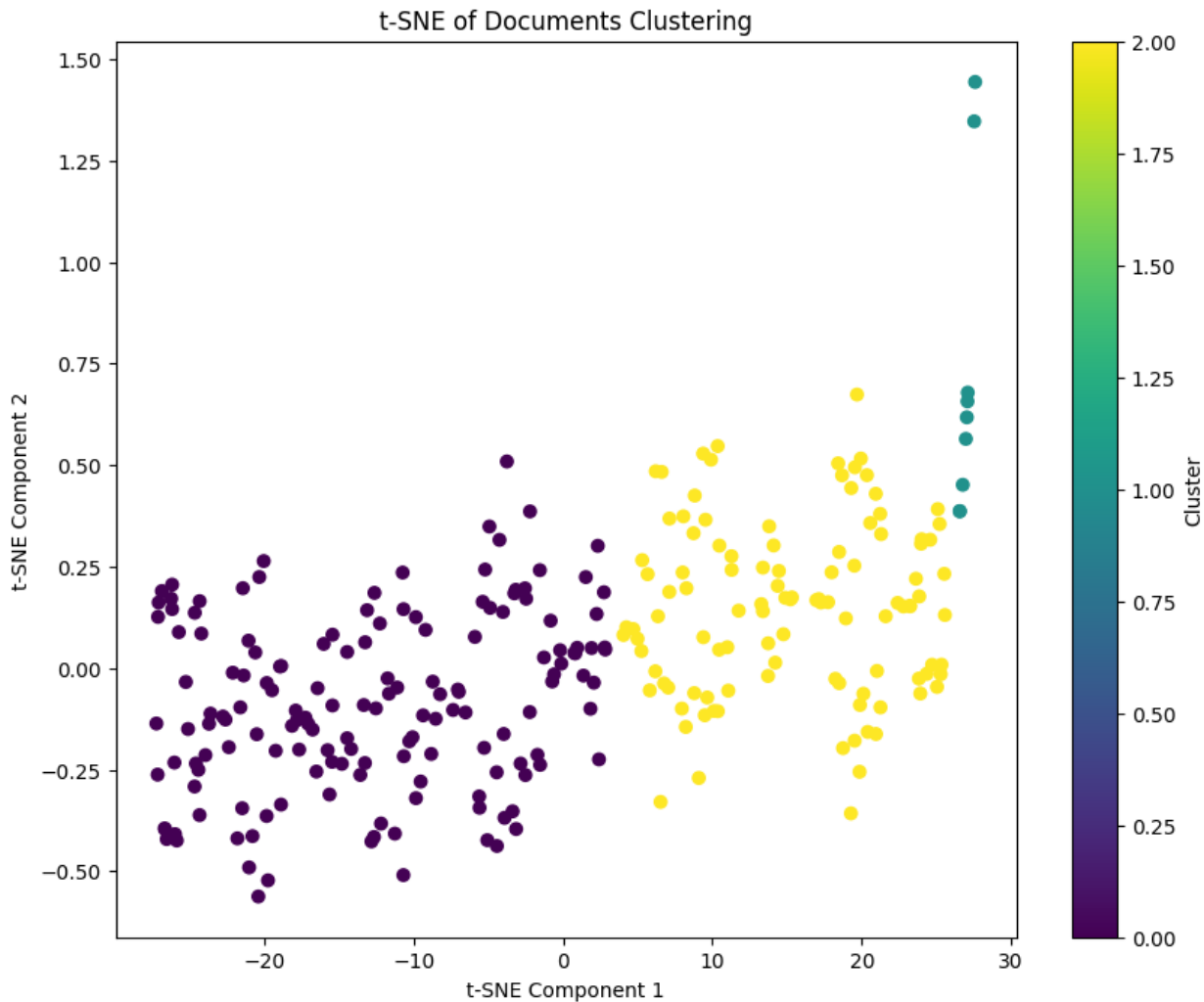


Figure 21:t-SNE plot

Complete visualization of document clusters offered by the above t-SNE plot. Each point of the cluster represents a document. Also, semantic similarity is reflected by its proximity to others. Thematic groupings such as news articles on policy changes, economic updates and stock performance described by distinct clusters. News articles that contain information about bullish sentiment and positive stock trends belong to cluster number one. While cluster two represents news articles related to negative sentiment such as economic downward and bearish investor sentiment. Cluster number three is focused on neutral content like market summaries, expert comments and company descriptions etc. The effectiveness of document vectorization and clustering can be identified by clear separation between clusters. Overlapping points within clusters may indicate mixed or transitional sentiment in the news articles. While t-SNE excels in maintaining local relationships, as seen by tight-knit clusters, its limitations in capturing global patterns imply that other visualizations, like PCA, may be essential to a deeper examination.

Interpretation of Results:

- Information's related to grouped documents and identified common themes within each cluster provided as insights.
- Most significant clusters and their relevance to the behavior of investors and market trend are highlighted here.

Dependencies:

- Python libraries: gensim, sklearn, matplotlib, seaborn, numpy.

Code Snippet:

```
t-SNE Visualization

[ ] from sklearn.manifold import TSNE

# Apply t-SNE for further dimensionality reduction
tsne = TSNE(n_components=2, random_state=42)
tsne_result = tsne.fit_transform(X)

# Plot the t-SNE results
plt.figure(figsize=(10, 8))
plt.scatter(tsne_result[:, 0], tsne_result[:, 1], c=df['Cluster'], cmap='viridis')
plt.title('t-SNE of Documents Clustering')
plt.xlabel('t-SNE Component 1')
plt.ylabel('t-SNE Component 2')
plt.colorbar(label='Cluster')
plt.show()
```

Figure 22:t-SNE Code

9. Dependency Parsing and Advanced Structures

Dependency parsing is an NLP for analyzing a sentence's grammatical structure through creating links between “head” words. And dependent phrases. This approach gives insights into the structure of syntactic words and enables for a more in-depth comprehension of text semantics.

Key Concepts:

- **Dependency Trees:**
 - This concept used tree architecture to represent the grammatical structure of a sentence. Words are acting as nodes of the tree while edges represent syntactic relationships.

Implementation Steps:

1. Dependency Parsing:

- To create dependency trees to represent the sentence structures Spacy and Stanford NLP libraries are used.
- For the process of capturing grammatical dependencies. Key relationships such as subjects, objects and modifiers are extracted.

2. Advanced Structures:

- Understand how words contributed to the overall meaning of a sentence semantic role labeling (SRL) was combined with dependency phrasing.
- Downstream tasks have been enhanced including question answering, relation extraction and syntax-aware sentiment analysis. By leveraging parsed structures.

3. Visualization:

- Dependency trees are used to generate visualizations of dependency trees by employing Spacy library's built-in plotting tools.
- As an Example: For the sentence "The market rise boosted investor confidence," Subject and object relationship, Market (subject), rise(object) and boosted (predicate) captured and visualized by the dependency tree.

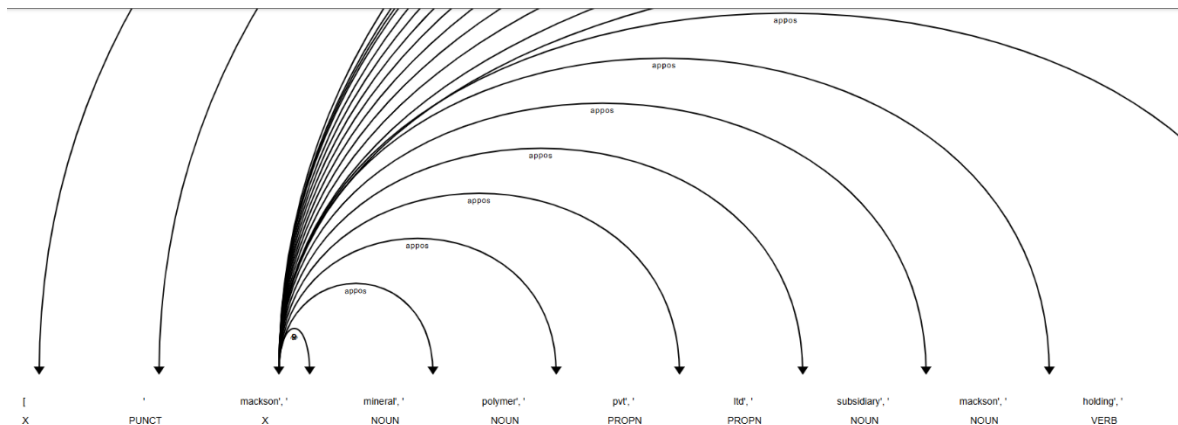


Figure 23: Dependency tree (Couldn't capture the full code because image is too lengthy)

Code Snippet:

Dependency Parsing and Advanced Structures

```
import spacy
import pandas as pd
from spacy import displacy

# Load SpaCy English model
nlp = spacy.load("en_core_web_sm")

# Assuming your DataFrame is 'df' and the 'Processed Text' column contains the text to be parsed
# Parse the first few rows for analysis
sample_texts = df['Processed Text'].head(5) # You can adjust the number of rows to sample

# Perform dependency parsing for each sentence (ensure text is passed as a string)
parsed_sentences = [nlp(str(text)) for text in sample_texts]

# Visualize the dependency parsing using displacy
for idx, parsed_doc in enumerate(parsed_sentences):
    print(f"Sentence {idx + 1}: {sample_texts.iloc[idx]}")
    displacy.render(parsed_doc, style='dep', jupyter=True)

# If you want to save the visualizations to HTML files
for idx, parsed_doc in enumerate(parsed_sentences):
    displacy.render(parsed_doc, style='dep', page=True, jupyter=False)
    with open(f"parsed_sentence_{idx + 1}.html", "w") as f:
        f.write(displacy.render(parsed_doc, style='dep', page=True, jupyter=False))
```

Figure 24: Dependency Phrasing and Plot Generation

Dependencies:

- Python libraries: spacy, stanza, networkx, matplotlib.

10. Insights and Real-World Applications

Objective:

Discuss real-world applications and key insights gained from the NLP analysis.

Key Insights:

1. Sentiment Trends and Market Behavior:

- Sentiment patterns that can impact on Colombo Stock Exchanges stock price actions was identified by analyzing sentiment trends from Sri Lankan news articles.
- Correlation between polarity of sentiment and stock market indexes was discovered. For instance, positive investor sentiment aligned with rising /bullish market trends, while negative sentiments of news articles corresponded to downtrends/bearish markets.

2. Topic Relevance:

- Imported topics such as corporate earnings, policy announcements and economic gains etc. Extracted topics are structured and classified as positive, negative or neutral and loaded into data frames.
- These insights from analysis are useful for the stakeholders to analysis the market climate before entering into the market.

3. Influence of Economic Events:

Correaltion of the news articles about major economic events and stock market price action was identified through this analysis. Significant sentiment changes can be affected on the market quickly such as central bank policy changes or political decisions. Investors' decision-making process can be leveled up.

1. Business Analytics:

- Sentimental analysis insights can be used to measure the public perception of market trends and customer interests. These insights are useful when creating business strategies and marketing campaigns.
- Example: sentiment about a product can be affected on its company's stock price. Hedge funds are using these business analytics concepts for their predictions.

2. Social Media and News Analysis:

- Using NLP insights to identify trending topics and consumer interests can be monitored through social media and news articles. Comments extraction from social media is a successful sentiment analyzing method. Effectiveness of social media insights is very high for political campaigns and marketing campaigns. Also, government security forces used this concept for their investigations about terrorism, racism and other illegal activities.
- Example: Facebooks sentiment from comment is used to analyze people's opinion about different parties.

3. Investment Strategies:

- This NLP based sentiment analysis can be used as a decision-making tool for investors. And these insights are acting as investment signals.
- Example: A sudden interest rate cut from central bank affected the market positively. Investors can buy shares at a lower price level before the news hits the mass audience of the investors.

4. Policy Research:

- Policy makers from government and semi-government organizations can use these insights, direct and indirect correlations from news to get more friendly decisions for investors and people
- Example: Monitoring the sentiment of people is used by government organizations to refine communication strategies around fiscal policy. "A sudden announcement of Sri Lankan banks losing their dollar bonds" can convert the market trend to a market crash. This kind of news needs to be published very carefully.

Impact on Decision-Making:

1. Enhanced Predictive Accuracy:

- Accuracy of market predictions was established by topic modeling and sentiment analyzing techniques. This allows stakeholders to anticipate sudden market reaction very effectively while maintaining their loss in a low range.

2. Improved User Behavior Understanding:

- Analyzing market dynamics is helpful when understanding customer preferences about purchasing, purchasing and customer satisfaction etc. Business intelligence concepts can apply to these insights to target loyal customers and target specific areas that have more customer interest.

3. Real-Time Insights:

- Actionable insights provided by the real time analysis by empowering investors, policy makers and journalists' Real-time analysis provides actionable insights that empower investors, journalists, and policymakers to make educated selection easily.

Summary:

- This highlights using NLP for transforming unstructured text data to useful ,meaningful and eye catching format is a must. Also, can generate actionable insights by using visualization concepts aligned with NLP. Also, this information's guide to make effective investment decisions, business decisions, and friendly policy developments.

11.Implementing Advanced NLP Techniques for text summarization

Objective:

Cutting edge NLP techniques have been used for text summarization. Text summarization and sentiment analysis generating insights for investor awareness. Transformers based advanced NLP concepts have been used to generate full summarizations.

Methodology:

1. Implementation Approach:

- The model was developed by employing transformers library that has capabilities to generate full text summarizations and highlight the key points from the row news from newspaper articles.
- For an effective and meaningful text summarization pretrained model called BART was employed.

2. Detailed Process:

- **Input Preparation:** Articles from newspapers paragraphs and text data was extracted in this stage of input preparation. In this stage of input preparation text data was fed into the program.

- **Tokenization:** Text data needs to be in a suitable format to feed the BART model. In this step transformers libraries tokenizer were employed for text tokenization. Tokenization is used to divide sentences into words.
- **Summarization:** BART model is generating critical information that benefits investors including significant themes, meaning full summarizations and sentiment analytics report for investors.
- **Output Refinement:** As the final output of this project insights from sentimental analytics and meaningful short summaries from news articles are displayed to investors to take their decisions.

Code Snippet:

```
import spacy
import pandas as pd

# Load SpaCy English model
nlp = spacy.load("en_core_web_sm")

# Initialize the relevant verbs list
relevant_verbs = {
    "boost", "win", "achieve", "announce", "grow", "expand", "raise", "reduce", "improve",
    "ranked", "increase", "gain", "surge", "lead", "dominate", "outperform", "capitalize",
    "invest", "strengthen", "transform", "optimize", "target", "secure", "partner", "introduce",
    "accelerate", "improve", "strengthen", "expand"
}

# Function to generate a short summary for each paragraph text
def summarize_paragraph(text):
    doc = nlp(text)

    summary = []
    for ent in doc.ents:
        if ent.label_ in {"ORG", "GPE", "EVENT"}: # Focus on organizations, geopolitical entities, and events
            summary.append(ent.text)

    # Extract relevant verbs
    actions = [token.lemma_ for token in doc if token.pos_ in {"VERB"} and token.lemma_ in relevant_verbs]

    if actions:
        summary.extend(actions)

    # Combine all important entities and actions into a summary
    return " ".join(summary) if summary else text

# Apply the summarization function to the 'Paragraph Text' column
summarized_texts = df['Paragraph Text'].apply(summarize_paragraph)

# Create a new DataFrame for summarized data
summarized_df = pd.DataFrame({
    'Original Text': df['Paragraph Text'],
    'Summarized Text': summarized_texts
})

# Display the summarized DataFrame
summarized_df
```

Figure 25: Summarization

Results and Significance:

- **Results:**

- The summaries were precise, retaining the essence of the original articles.
- Investors can drive their decisions by observing analytical insights such as highlighted trends, sentiments and investor consultation.

- **Significance in Project Context:**

- Raw textual data is converted to actionable insights and bridges the gap between raw data and actionable intelligence by summarization.
- Time efficiency is ensured by this concept while ensuring that critical information is readily available in investors finger point. Also, this project creates a more secure environment for investors who are investing in stock markets.
- This advanced NLP approach demonstrates the capability of the project to harness modern tools for real-world problems, aligning with the goal of delivering high-impact and reliable data solutions.

```
Improved Business Sector Summary (User-Friendly):
Cluster 0: [CENTRAL BANK] increase. Sentiment: Positive
=====
Cluster 2: [COPILOT] announce. Sentiment: Neutral
=====
Cluster 2: [ASUS] announce. Sentiment: Neutral
=====
Cluster 2: [INTEL] announce. Sentiment: Neutral
=====
Cluster 2: [NPU] announce. Sentiment: Neutral
=====
Cluster 0: [OXFORD COLLEGE BUSINESS] grow secure. Sentiment: Positive
=====
Cluster 2: [BALMORAL HALL KINGSBURY HOTEL COLOMBO] partner. Sentiment: Positive
=====
Cluster 2: [KASPERSKY] partner. Sentiment: Positive
=====
Cluster 2: [INSIGHTEDGE] increase. Sentiment: Positive
=====
Cluster 2: [CIC] introduce. Sentiment: Positive
=====
```

Figure 26: Highlighted summary With ORG names

```
Overall, the market sentiment is positive. Investors may consider capitalizing on opportunities with growing organizations.
```

Figure 27: Investment Consultations

Expected Outcomes:

1. **Enhanced Information Accessibility:** This approach of summarization provides an investor friendly environment for investors to make informed decisions on the stock market without reading lengthy news articles.
2. **Improved Decision-Making:** Enhanced meaningful summaries and sentiment analysis reports and investor consultation about total sentiment enables a better focus on trends and opportunities in a volatile market.
3. **Scalability:** This project can be enhanced to process multiple languages and can be expanded to deal with real time data from APIs.

12. Conclusion

Transformative power of Natural Language Processing is revealed by the extracting process of unorganized data in this project while providing actionable and reliable insights from row text data. We provided tools to retrieve data for investors to get investment decisions, details for policy makers and sentiment analysis and market trends businesses.

This approach of NLP describes the importance of sentiment analysis for understanding volatile markets, Capture market trends and effective buying or selling opportunities. Also, the effectiveness of topic modeling in extracting recurring themes. A deeper understanding about syntactic and semantic relationships provided by dependency phrasing and advanced NLP structures.

An advanced sentiment analysis and market forecasting task was added as the bonus task to summarize big, scaled data. The potential need of advanced NLP tasks to address the real-world problems was highlighted by this project. The bonus task on text summarization added another layer of utility, providing concise yet comprehensive summaries of lengthy articles. The potential importance of NLP concepts to solve real world challenges is highlighted by this approach. Challenges such as training in volatile markets, analyzing reader sentiment and how much impact media can have on stock markets etc.

As the Future directions for this project real time data analyzing features, time series predictions align with NLP, adding neural networks for text classification and need to add multilanguage processing capabilities need to integrate because Sinhala is the primary language in Sri Lanka. Also, for advanced level performance GPT models can be integrated with this to ensure accuracy and robustness.

Finally this project plays a critical role in Natural Language Processing by converting unstructured and row data to actionable insights for investors while providing reliable and quality analytical information for investors to make better decisions.

References

1. Bergsma Shane, ,. M. (2012). Stylometric Analysis of Scientific Articles. *Conference of the North American Chapter of the Association for Computational Linguistics*, 327–337.
2. Linkai Zhu, M. H. (2010). A N-gram based approach to auto-extracting topics from. 344-355.
3. Setiawan, Y. (2024). The Optimization of n-Gram Feature Extraction Based on Term Occurrence for Cyberbullying Classification. *Journal of data science*, 41-47.