Web Scraping and Sentiment Analysis of Financial News Using Python

Kisha Prajapati 21D070048
Electrical Engineering, IIT Bombay
Email: 21D070048@iitb.ac.in

Akshat Taparia 210110014
Electrical Engineering, IIT Bombay
Email: 210110014@iitb.ac.in

Tanishka Pradhan 210040159
Electrical Engineering, IIT Bombay
Email: 210040159@iitb.ac.in

***Abstract*—This paper presents a complete workflow for collecting, cleaning, and analyzing financial news headlines using Python-based web scraping and Natural Language Processing (NLP). News data was scraped from CNBC, Reuters, and The Guardian using BeautifulSoup. After cleaning more than 50,000 headlines, sentiment analysis was performed using TextBlob and VaderSentiment, followed by clustering with K-Means, Hierarchical Clustering, and Jenks Natural Breaks Optimization. The goal is to classify financial news into positive, neutral, and negative sentiment categories. The methodology and experimental results demonstrate how automated news sentiment extraction can support financial forecasting and market analytics.**

***Index Terms*—Web scraping, BeautifulSoup, sentiment analysis, NLP, TextBlob, VaderSentiment, clustering, financial news.**

I. Introduction

The rapid expansion of digital journalism and the growing influence of online financial media have reshaped how investors, analysts, and automated trading systems interpret real-time market information. Financial news directly impacts asset prices, investor psychology, and short-term market volatility. With major news outlets such as CNBC, Reuters, and The Guardian publishing thousands of articles every month, there is an urgent need for automated systems capable of extracting, processing, and analyzing large-scale textual data efficiently. This necessity positions web scraping and Natural Language Processing (NLP) as essential tools for modern financial analytics.

In this work, we develop a comprehensive pipeline that begins with large-scale web scraping of financial news headlines and article descriptions using Python's `BeautifulSoup` library. The scraping process targets three high-volume sources—CNBC, Reuters, and The Guardian—each of which presents unique structural and metadata challenges in their HTML layouts. As demonstrated in the collected notebook data, custom scraping logic, exception handling, and iterative pagination techniques were required to reliably capture headlines, timestamps, and descriptions across multiple page formats. Reuters alone contains more than 3,000 archived pages, while The Guardian uses dual HTML blocks per page and omits article preview text entirely. These inconsistencies highlight the importance of designing flexible and fault-tolerant scraping functions for real-world web data.

Following data collection, extensive preprocessing steps are performed. These include removing duplicate entries, eliminating missing values, correcting inconsistently formatted timestamps, and normalizing month abbreviations such as "Sept" and full month names observed in CNBC's dataset. All timestamps are converted into a uniform `datetime64` format, allowing for accurate temporal comparisons and trend analysis. After cleaning, the final datasets consist of approximately 2,790 CNBC articles, 32,696 Reuters articles, and 17,794 Guardian articles, forming a strong textual foundation for sentiment analysis.

To quantify sentiment in financial headlines, two established NLP tools are applied: TextBlob and VaderSentiment. TextBlob generates polarity scores between -1 and $+1$, capturing the emotional tone of both headlines and descriptions. VaderSentiment provides compound scores that integrate positive, neutral, and negative components, and is particularly effective for short financial text. Exploratory analysis of these polarity distributions reveals significant noise, overlap, and imbalance across articles, indicating that raw sentiment scores are insufficient for direct sentiment classification.

To address this challenge, unsupervised machine learning methods are employed. Specifically, K-Means clustering, Agglomerative Hierarchical Clustering, and Jenks Natural Breaks Optimization are used to group articles into positive, neutral, and negative sentiment categories. These clustering algorithms operate on multidimensional polarity vectors, allowing the model to separate sentiment groups even when polarity scores overlap heavily. Visualizations of cluster boundaries demonstrate clear structural differences in sentiment distribution across sources, and highlight which clustering method performs best for each dataset.

The final output of this research is a set of sentiment-labeled datasets derived from both TextBlob and Vader scoring methods. These labeled datasets serve as valuable resources for downstream applications such as market trend prediction, volatility forecasting, algorithmic trading strategies, and sentiment-aware financial dashboards.

Overall, this work demonstrates a fully integrated framework that combines web scraping, data engineering, NLP sentiment extraction, and unsupervised learning. By applying these methods to large-scale financial news, the study contributes a scalable and replicable model for automated sentiment analysis, offering practical value to financial researchers, data scientists, and trading system developers.

This paper contributes:

- A scalable Python scraping framework for news websites.
- Cleaning and normalization of inconsistent timestamps.
- Sentiment extraction using two NLP methods: TextBlob and Vader.
- Clustering-based sentiment classification.

II. Web Scraping Methodology

This study employs a robust and scalable web scraping framework to collect financial news headlines and article metadata from three major news publishers: CNBC, Reuters, and The Guardian. All scraping was implemented in Python using the `requests` and `BeautifulSoup4` libraries. Due to structural variations in the HTML layout of each source, customized extraction logic, dynamic pagination, and exception-handling mechanisms were necessary to ensure completeness and accuracy of the scraped datasets.

A. Tools and Libraries Used

The web scraping pipeline utilizes the following Python libraries:

- **requests**: Retrieves HTML content via HTTP GET requests.
- **BeautifulSoup4**: Parses HTML structures and extracts relevant elements.
- **pandas**: Stores extracted data in tabular form for cleaning and analysis.
- **time**: Handles delays between page requests to avoid overloading servers.

B. General Scraping Workflow

All three datasets were collected using a consistent scraping workflow:

1) Send a GET request to each page URL.
2) Parse the HTML document using BeautifulSoup's `html.parser`.
3) Extract headlines, timestamps, and descriptions using tag-based filtering.
4) Append extracted fields to a `pandas` DataFrame.
5) Iterate through paginated URLs until no more articles are available.
6) Save the complete dataset to CSV for further cleaning and sentiment analysis.

This workflow ensures reproducibility, modularity, and fault tolerance across all three sources.

C. Scraping CNBC

CNBC's S&P 500 news archive was scraped across 140 sequential pages. For each page, articles are located inside the HTML container:

```
<div class="Stories-lineup bigHeader">
```

From each article block, the following fields were extracted:

- **Headline:** Extracted from the `<a>` tag within the article block.
- **Timestamp:** Retrieved from the `<time>` tag containing publication date.
- **Description:** Extracted from the summary `<div>` when available.

CNBC's timestamps include multiple inconsistent month formats (e.g., "Sept", "March", "Jul"), which required normalization in later cleaning steps. Custom `try/except` blocks were implemented to handle missing descriptions and irregular article structures.

D. Scraping Reuters

Reuters provides the largest dataset, with more than 3,200 pages in its archived "Business News" section. Each page contains 10 articles and follows the URL structure:

```
?page=n&view=page&pageSize=10
```

Articles are contained in:

```
<article class="story">
```

For each article, the scraper extracted:

- **Headline**
- **Timestamp**
- **Article Description**

Reuters provides cleaner HTML compared to CNBC, but variations in timestamp structure (e.g., missing year, relative timestamps like "2 hours ago") required additional post-processing.

E. Scraping The Guardian

The Guardian uses a two-block repeated structure per page, with articles embedded within:

```
<div class="fc-container__inner">
```

Unlike CNBC and Reuters, The Guardian does not provide article descriptions on listing pages. Therefore, only the following were extracted:

- **Headline**
- **Timestamp**

Two separate HTML containers per page were scraped to ensure full coverage. Inconsistent timestamp formatting (e.g., "2d ago", "14h ago") required conversion to absolute datetime values during cleaning.

F. Error Handling and Robustness

Due to inconsistent HTML structures, frequent missing fields, and dynamically generated elements, multiple robustness techniques were used:

- **Conditional extraction**: Avoids crashes when fields are absent.
- **URL pagination loops**: Stops scraping automatically when no further articles are found.
- **Polite scraping delays**: `time.sleep()` is used to avoid rapid consecutive requests.
- **HTML fallback handling**: If primary tags fail, alternative tags are checked.

These measures ensured that the scraper collected clean, complete datasets without interruption.

G. Final Dataset Summary

The final scraped datasets consist of:

- CNBC: approximately 2,790 articles
- Reuters: approximately 32,696 articles
- The Guardian: approximately 17,794 articles

These datasets form the input for subsequent cleaning, normalization, exploratory analysis, polarity scoring, and clustering-based sentiment classification.

III. Data Cleaning Methodology

Following the web scraping stage, the collected datasets from CNBC, Reuters, and The Guardian underwent an extensive data cleaning process to ensure consistency, accuracy, and suitability for downstream sentiment analysis. The raw scraped data contained missing values, duplicated entries, inconsistent timestamp formats, and heterogeneous date representations, as demonstrated in the notebook :contentReference[oaicite:1]index=1. This section describes the complete cleaning workflow applied to each dataset.

A. Overview of Cleaning Steps

The data cleaning pipeline consisted of the following major steps:

1) Removal of incomplete or missing records.
2) Deduplication of repeated articles based on headline or headline–description pairs.
3) Normalization of timestamp formats across all datasets.
4) Resolution of non-standard month names and abbreviations.
5) Parsing and restructuring of date and time fields into machine-readable formats.
6) Reordering and structuring cleaned fields into consistent tabular form.

Each of these steps was required due to the structural inconsistencies observed across the scraped datasets.

B. Cleaning CNBC Data

The CNBC dataset initially contained numerous rows with missing values in the `Headlines`, `Time`, or `Description` fields. After loading the dataset, cleaning involved:

- **Dropping rows with missing values**: Ensuring completeness of textual and temporal information.
- **Removing duplicates**: Since multiple CNBC pages reuse identical article blocks, duplicates were removed using the pair `(Headlines, Description)`.

A unique challenge arose from CNBC's inconsistent datetime conventions. The scraped timestamps used non-standard month strings such as "Sept", and sometimes full month names like "March", "April", or "July", as shown in the notebook output (Page 6) :contentReference[oaicite:2]index=2. To correct this, a custom normalization function was implemented:

```
def replace_dt(s):
    s = s.replace("Sept","Sep")
    .replace("March","Mar")
    s = s.replace("April","Apr")
    .replace("June","Jun")
    s = s.replace("July","Jul")
    if s[0].isspace():
        s = s.replace(" ", "0", 1)
        s = s.replace(", ", ", 0", 1)
    return s
```

This function standardized all month names and padded single-digit hours and dates with leading zeros. After replacement, all CNBC timestamps were parsed using the format:

```
%I:%M %p ET %a, %d %b %Y
```

The parsed output was separated into:

- A **Date** column (as `datetime64`)
- A **Time** column (24-hour format)

The cleaned CNBC dataset contained 2790 complete and non-duplicated rows.

C. Cleaning Reuters Data

Reuters data included both headlines and article descriptions but had a simpler timestamp structure. The primary cleaning tasks were:

- **Dropping any missing rows** in the scraped dataset.
- **Removing duplicate headline–description pairs**.
- **Converting the `Time` column directly into datetime format**.

Reuters timestamps used the format:

```
Mon DD YYYY
```

This allowed direct parsing into a single **Date** column. No additional custom functions were needed since Reuters uses standardized month abbreviations consistently. The cleaned Reuters dataset contained 32,696 valid articles, covering data from March 2018 onward.

D. Cleaning The Guardian Data

The Guardian dataset differed substantially from the others because:

- It does **not** include article descriptions.
- Each page contains two distinct HTML containers for articles, leading to structural repetition.

Cleaning steps included:

- **Dropping missing rows**.
- **Removing duplicates** based solely on the `Headlines` field.
- **Parsing the `Time` column**, which used the format `DD Month YYYY`

For example: `19 July 2020` → converted into a `datetime64[ns]` **Date** column.

The cleaned Guardian dataset consisted of 17,794 unique entries.

E. Final Structuring and Storage

After cleaning all three datasets independently, each dataset was reorganized into a consistent column structure to facilitate uniform downstream processing. For example:

- CNBC: `[Date, Time, Headlines, Description]`
- Reuters: `[Date, Headlines, Description]`
- Guardian: `[Date, Headlines]`

All cleaned datasets were stored for later use in exploratory analysis, polarity computation, and clustering:

```
%store df1   # CNBC
%store df2   # Reuters
%store df3   # Guardian
```

These standardized, fully cleaned datasets formed the foundation for the sentiment analysis and machine learning components of the study.

IV. Sentiment Analysis Methodology

To quantify the emotional tone contained in the financial news headlines and article previews, this study employs two widely used Natural Language Processing (NLP) sentiment analysis tools: **TextBlob** and **VaderSentiment**. Both tools were applied to the cleaned datasets described in the previous section. The goal of this analysis is to compute polarity scores that capture the degree of positivity, negativity, or neutrality present in each article. These polarity metrics were later used as inputs for clustering-based sentiment classification. All procedures described in this section are implemented in the accompanying notebook :contentReference[oaicite:1]index=1.

A. Overview of Tools Used

- **TextBlob**: A rule-based NLP library that computes a polarity score in the range $[-1, 1]$, where -1 represents strong negativity, $+1$ represents strong positivity, and values near 0 indicate neutrality.
- **VaderSentiment**: A lexicon- and rule-based model optimized for short text (e.g., news headlines, tweets). Vader outputs four measures: `neg`, `neu`, `pos`, and a final **compound score**, which lies in the range $[-1, 1]$ and reflects an aggregated sentiment value.

Because news headlines are concise and context-driven, combining both tools allows for more reliable sentiment interpretation.

B. TextBlob Polarity Extraction

TextBlob was first applied to the **Headlines** and **Descriptions** (when available) for all three datasets: CNBC, Reuters, and The Guardian. For each text entry, the following command was executed:

```
wiki = TextBlob(value)
polarity = wiki.sentiment.polarity
```

This generated two polarity columns for multi-field datasets (CNBC and Reuters):

- `tb_hl_polarity`: headline polarity
- `tb_ds_polarity`: description/preview polarity

For The Guardian dataset, which contains headlines only, a single column (`tb_hl_polarity`) was produced. Examples of these polarity values are shown in the notebook (e.g., Page 14 and Page 16) :contentReference[oaicite:2]index=2.

Scatter plots of headline-versus-description polarity revealed highly overlapping clusters and substantial noise, indicating that raw TextBlob polarity was insufficient to directly classify sentiment. Therefore, clustering techniques were required in later stages.

C. VaderSentiment Polarity Extraction

VaderSentiment was applied in a similar fashion, but the focus was primarily on the **compound score**, which provides a normalized, aggregated sentiment score:

```
analyzer = SentimentIntensityAnalyzer()
compound = analyzer.polarity_scores(value)['compou
```

As shown in the notebook (Page 30–36) :contentReference[oaicite:3]index=3, two compound polarity features were extracted for CNBC and Reuters:

- `vs_hl_compounds`: headline compound score
- `vs_ds_compounds`: description compound score

For The Guardian dataset, only the headline compound score was calculated due to the absence of article descriptions.

Just as with TextBlob, Vader compound scores showed significant variance and cross-distribution between positive, neutral, and negative regions. Scatter plots of headline versus description compound values revealed distinct but overlapping clusters, confirming the need for unsupervised learning techniques for grouping.

D. Visualization and Initial Observations

To assess the structure of the polarity distributions, scatter plots and histograms were generated for each dataset. These visualizations demonstrated:

- Both TextBlob and VaderSentiment produce polarity values roughly centered around zero.
- Many articles exhibit conflicting headline and description polarity, especially for Reuters where descriptions often soften or contrast with headline tone.
- The polarity space is continuous and lacks clear linear boundaries, making threshold-based sentiment labeling unreliable.

Due to this noise and overlap, the study proceeded with **unsupervised clustering** techniques (K-Means, Hierarchical Clustering, and Jenks Natural Breaks Optimization) to separate articles into sentiment categories. These clustering methods are described in the following section.

E. Data Storage for Clustering

After computing polarity and compound scores, all datasets were stored in structured form for downstream clustering analysis:

```
%store textblob_df1
%store textblob_df2
%store textblob_df3
%store vader_df1
%store vader_df2
%store vader_df3
```

These stored datasets include both raw polarity values and polarity vectors used for clustering (e.g., {tb_hl_polarity, tb_ds_polarity}, {vs_hl_compounds, vs_ds_compounds}).

F. Summary

The sentiment analysis methodology established a dual-layer polarity extraction framework using TextBlob and VaderSentiment. TextBlob provided a general linguistic polarity measure, while VaderSentiment introduced a compound score better suited for short financial text. The combination of both tools produced four-dimensional sentiment vectors for multi-field datasets and one-dimensional sentiment values for headline-only datasets.

These polarity metrics served as the foundation for the sentiment clustering models developed in the next section.

V. Results

This section presents the empirical findings obtained from the web-scraped financial news datasets sourced from CNBC, Reuters, and The Guardian. The results summarize article volume patterns, polarity distributions, clustering-based sentiment classification, and comparative observations across the three publishers. All analyses are based on the processed datasets and sentiment computations described in earlier sections, with workflows and plots derived directly from the notebook :contentReference[oaicite:1]index=1.

A. Article Volume Analysis

Across the three publishers, a total of 53,280 news articles were collected between 2017 and 2020. Monthly article-frequency plots show consistent publication patterns within each dataset:

- **CNBC:** Article volume fluctuates between 40 and 120 articles per month (see Page 12). Peaks correspond to market-moving events such as earnings seasons and macroeconomic announcements.
- **Reuters:** A significantly larger volume, ranging between 800 and 1,800 articles per month (Page 13), reflecting Reuters' role as a global wire service.
- **The Guardian:** Article counts generally range from 300 to 750 articles per month, with pronounced surges during major economic or political developments (Page 14).

These temporal publication patterns confirm dataset robustness and provide a solid foundation for sentiment-based time series analysis in future work.

B. TextBlob Polarity Results

TextBlob polarity distributions indicate substantial noise and overlap across all datasets:

- **CNBC:** Scatter plots of headline versus description polarity (Page 19) show no clear linear separation. Positive headlines often accompany negative previews, producing conflicting sentiment signals.
- **Reuters:** A similar pattern emerges (Page 22), where large clusters of articles concentrate near zero polarity, indicating neutral or mixed sentiment.
- **The Guardian:** As this dataset includes headlines only, TextBlob polarity forms a broad but unimodal distribution centered around zero (Page 23), suggesting higher linguistic neutrality.

Since polarity scores alone did not sufficiently distinguish sentiment classes, clustering was performed as a necessary next step.

C. Clustering-Based Sentiment Classification (TextBlob)

1) CNBC: Using K–Means clustering on the two-dimensional polarity vectors (*headline*, *description*), three clusters emerged (Page 25):

- A negative cluster concentrated in the bottom-left region.
- A positive cluster in the upper-right quadrant.
- A neutral cluster scattered between the two extremes.

Post-inspection labeling assigned cluster identities as:

$$\text{Negative} = 0, \quad \text{Neutral} = 1, \quad \text{Positive} = 2$$

2) Reuters: Reuters polarity clustering via K–Means similarly produced three sentiment groups (Page 27). Visual inspection revealed:

- Blue points representing negative sentiment,
- Green points representing positive sentiment,
- Red points forming a neutral cluster.

The final mapping for Reuters was:

$$\text{Negative} = 0, \quad \text{Neutral} = 1, \quad \text{Positive} = 2$$

3) The Guardian: Since this dataset included only one polarity dimension, Jenks Natural Breaks Optimization was applied (Page 28). Resulting breakpoints:

$$[-1.0,\ -0.225,\ 0.208,\ 1.0]$$

yielded clear separation into:

$$\text{Negative} = x \leq -0.225, \quad \text{Neutral} = -0.225 < x < 0.208, \quad \text{Positive} =$$

D. VaderSentiment Results

Vader compound scores showed stronger sentiment clustering compared to TextBlob due to their design for short, news-style text.

1) CNBC: Hierarchical clustering yielded better separation than K–Means (Page 39). Clusters were interpreted as:

$$\text{Positive} = 2, \quad \text{Negative} = 1, \quad \text{Neutral} = 0$$

with the negative class forming a dense horizontal band and positive articles concentrated in the upper-right.

2) Reuters: Because of memory constraints, only K–Means was used (Page 41). Cluster inspection showed:

- Blue: positive articles,
- Red: negative articles,
- Green: neutral articles.

3) The Guardian: Only headline compounds were available. The distribution (Page 36) showed clear left-skewness toward negative sentiment. Based on the histogram, empirical breakpoints were selected to classify sentiment into negative, neutral, and positive categories.

E. Overall Sentiment Distribution

Combining TextBlob and VaderSentiment classifications yields the following overarching trends:

- **Negative headlines dominate** across all three publishers, particularly for Reuters, which frequently reports crisis-related news.
- **Neutral sentiment forms the second-largest group**, reflecting informational, non-emotional reporting style common in financial journalism.
- **Positive sentiment is least common**, appearing mainly in earnings beats, economic recoveries, and corporate success stories.

These patterns align with expected characteristics of business journalism, where market downturns, risks, and economic uncertainty typically draw more coverage.

F. Key Observations

The results across all three datasets demonstrate the following:

1) Short financial headlines often contain mixed sentiment, making dual-field polarity analysis (headline and description) essential.
2) Clustering was necessary because raw polarity values lacked clear decision boundaries.
3) VaderSentiment provided more stable grouping compared to TextBlob.
4) The Guardian, lacking descriptions, required one-dimensional optimization.
5) Reuters' extensive dataset enabled highly reliable statistical patterns.

Overall, the sentiment classification pipeline effectively separated negative, neutral, and positive news articles despite inconsistencies and noise in raw polarity values.

VI. Conclusion

This study presented a complete end-to-end framework for the automated extraction, cleaning, sentiment analysis, and categorization of large-scale financial news articles from CNBC, Reuters, and The Guardian. Through the integration of web scraping, data preprocessing, polarity computation, and unsupervised clustering techniques, the research demonstrates how heterogeneous online financial text can be transformed into structured and sentiment-labeled datasets suitable for further quantitative analysis.

The web scraping methodology successfully collected more than 53,000 articles using tailored extraction logic and dynamic pagination. This established a diverse, multi-source dataset containing both headlines and article previews where available. The subsequent data cleaning pipeline resolved inconsistencies caused by missing values, duplicated entries, noisy timestamps, and irregular HTML structures. By standardizing date formats and organizing each dataset into uniform tabular schemas, the study ensured high data integrity for downstream processing.

Sentiment analysis using both TextBlob and VaderSentiment revealed that raw polarity values exhibit substantial overlap and lack clear thresholds for direct class separation. Scatter plots, polarity histograms, and comparative visualizations further highlighted the continuous nature of sentiment distributions within financial journalism. This confirmed the necessity for clustering-based sentiment segmentation rather than simple rule-based approaches.

To address this challenge, three clustering methods—K–Means, Agglomerative Hierarchical Clustering, and Jenks Natural Breaks Optimization—were implemented across the datasets. These models effectively partitioned the polarity space into three coherent sentiment groups: negative, neutral, and positive. Reuters and CNBC benefited from two-dimensional polarity vectors (headline and description), while The Guardian required one-dimensional clustering due to the absence of article previews. Cluster interpretation and normalization transformed these outputs into consistent sentiment labels across all datasets.

Several key insights emerged from the final sentiment distributions. Negative news dominated all three datasets, reflecting the nature of financial reporting, in which adverse events such as market declines, economic crises, or geopolitical risks tend to generate high coverage. Neutral sentiment also remained substantial, indicating the prevalence of factual, event-driven reporting common in business journalism. Positive sentiment was the least represented, typically corresponding to corporate earnings, recovery indicators, or business expansions.

Overall, the results demonstrate that combining rule-based sentiment extraction with unsupervised clustering enables more reliable classification of financial news sentiment than using polarity measures alone. The methodology developed in this work provides a scalable and reproducible foundation for future applications such as market sentiment tracking, algorithmic trading, risk analysis, and predictive financial modeling. Future work may extend the framework by incorporating deep learning–based sentiment models, topic modeling, or time-aware sentiment forecasting to capture the evolving emotional tone of financial news in real time.

References

[1] T. Loughran and B. McDonald, "When Is a Liability Not a Liability? Textual Analysis, Dictionaries, and 10-Ks," *Journal of Finance*, vol. 66, no. 1, pp. 35–65, 2011.

[2] P. C. Tetlock, "Giving Content to Investor Sentiment: The Role of Media in the Stock Market," *The Journal of Finance*, vol. 62, no. 3, pp. 1139–1168, 2007.

[3] C. J. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text," *Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media*, 2014.
[4] S. Loria, "TextBlob: Simplified Text Processing," *TextBlob Documentation*, 2018. [Online]. Available: https://textblob.readthedocs.io
[5] R. Schumaker and H. Chen, "Evaluating a News-Aware Quantitative Trader: The Effect of Momentum and Contrarian Indicators," *ACM Transactions on Information Systems*, vol. 30, no. 2, pp. 1–23, 2012.
[6] C. W. Calomiris and H. Mamaysky, "How News and Its Context Drive Risk and Returns Around the World," *Journal of Financial Economics*, vol. 144, no. 2, pp. 507–551, 2022.
[7] G. Reis, A. Storkey, and N. Firoozye, "Modeling Market Impact with Deep Learning," *Proceedings of the 2019 ICAIF ACM International Conference on AI in Finance*, 2019.
[8] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep Learning for Event-Driven Stock Prediction," *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.
[9] S. Kogan et al., "Predicting Risk from Financial Reports with Regression," *NAACL Conference on Human Language Technologies*, pp. 272–280, 2009.
[10] G. F. Jenks, "The Data Model Concept in Statistical Mapping," *International Yearbook of Cartography*, vol. 7, pp. 186–190, 1967.
[11] S. Lloyd, "Least Squares Quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.
[12] F. Murtagh and P. Legendre, "Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion?," *Journal of Classification*, vol. 31, pp. 274–295, 2014.
[13] J. Boudoukh, R. Feldman, S. Kogan, and G. Richardson, "Which News Moves Stock Prices? A Textual Analysis," *Review of Financial Studies*, vol. 32, no. 5, pp. 2023–2053, 2019.