News Sentiment Analysis for Predicting Next-Day Stock Movements
CS 6120 – Final Project Report

Team Members:
Abhinav Anil Kumar (anilkumar.ab@northeastern.edu),
Shreepoorna Purohit(purohit.shr@northeastern.edu)

CS 6120 – Natural Language Processing

This report details our investigation into predicting next-day stock movements through news sentiment analysis. We compared two main methodologies: sentiment-based prediction, which utilized pre-trained models like VADER and FinBERT, and direct text-based prediction using transformer models, specifically employing FinBERT embeddings. The report includes an analysis and comparison of the results from both approaches to assess their efficacy in forecasting stock market fluctuations.

2. Grade Contract Milestones and Completion Status

Our grade contract specified a series of milestones establishing the scope and depth of our project work. Below, we describe each milestone and evaluate whether it was achieved, partially achieved, or not achieved, along with brief explanations.

Milestone 1: Data Preparation & Cleaning — Status: Achieved
We successfully extracted a multi-year subset of the FNSPID dataset (2018–2024) and filtered it to the 10 major U.S. stocks selected for study. We parsed timestamps, removed invalid entries, handled duplicate records, and standardized formats. We merged news items with corresponding stock price data for each ticker and computed next-day returns using OHLCV values. This processed dataset became the foundation for all subsequent experiments.

Milestone 2: Label Construction (Up / Down / NoChange) — Status: Achieved
To construct the labels for market movement, we employed a **deterministic labeling scheme** based on the **next-day return** calculated using stock price data. Specifically, for each news article, we followed these steps:

1. **Next-Day Return Calculation:**
   The **next-day return** for each stock was computed as the percentage change in the closing price from one day to the next:

$$\text{next\_day\_return} = \frac{CLOSE(t+1) - CLOSE(t)}{CLOSE(t)}$$

   where CLOSE(t+1) : is the closing price of the stock on the following day t+1
   CLOSE(t) : is the closing price of the stock on day t

   This calculation was applied to all stock price data to derive a **next-day return** for each stock.

2. **Labeling Scheme:**
   The next-day return was used to assign one of three labels to each article:

- **Up**: If the **next-day return** ≥ +0.2%, indicating a positive market movement,
- **Down**: If the **next-day return** ≤ −0.2%, indicating a negative market movement,
- **NoChange**: If the **next-day return** was between -0.2% and +0.2%, indicating little to no market movement.

This deterministic scheme allowed us to categorize market movements into the three classes.

3. **Training and Evaluation Splits**:

We implemented **time-based splits** to ensure **realistic evaluation** and to prevent **temporal leakage**:

- **Training Set:** 2018–2021 data (used to train the models).
- **Validation Set:** 2022 data (used to evaluate model performance during training).
- **Test Set:** 2023–2024 data (used to evaluate model generalization on unseen data).

This time-based approach ensured that the model was trained on past data and evaluated on future data, mimicking a real-world scenario where stock movement predictions are made using historical news articles and evaluated on subsequent market behavior.

Milestone 3: Baseline Sentiment Models — Status: Achieved

We implemented sentiment-based baseline models using **VADER** and **FinBERT** sentiment scores (positive/neutral/negative). These features were used to train **Logistic Regression (LR)** and **Random Forest (RF)** classifiers to predict market movement (Up/Down/NoChange).

- **Results:**
  - **VADER + LR:**
    - **Accuracy:** 49.39%, **Macro F1:** 22.08%
  - **VADER + RF:**
    - **Accuracy:** 46.79%, **Macro F1:** 30.12%
  - **FinBERT + LR:**
    - **Accuracy:** 45.73%, **Macro F1:** 31.98%
  - **FinBERT + RF:**
    - **Accuracy:** 46.63%, **Macro F1:** 32.11%

These results show that **Random Forest** models generally outperformed **Logistic Regression**, and **FinBERT-based models** slightly exceeded **VADER-based models**, providing richer features for stock movement prediction.

Milestone 4: FinBERT Embedding-Based Models — Status: Achieved

We used **FinBERT** to generate **mean-pooled embeddings** from news headlines, and later combined **headline + body text** for improved representation. The **tokenization** process involved splitting the text into tokens, which were then passed through **FinBERT** to produce **768-dimensional embeddings** for each article. These embeddings were used as input features for training classical ML models, such as **Logistic Regression** and **Random Forest**.

This approach significantly enhanced model expressiveness by capturing deeper contextual features of the text, going beyond sentiment polarity to leverage the full textual content.

Milestone 5: Web Scraping for Full Article Text — Status: Achieved

We identified a limitation in the **FNSPID dataset**, where the **body** field was often **truncated or missing**. To address this, we implemented a custom scraping pipeline using **newspaper3k** to retrieve full article text for URLs in the dataset.

Web Scraping Process:

- Articles were scraped by fetching their **full text** using the URLs present in the dataset.
- **Challenges:** Some sites had restrictions, requests failed, or time constraints hindered scraping for certain URLs.

Scraping Results:

- Full text was successfully retrieved for **100%** of the articles in the dataset, amounting to **56,506 articles**.
- This scraping effort greatly enhanced our ability to train and evaluate **full-text models** and provided a richer dataset for subsequent experiments.

This milestone allowed us to test full-text models and significantly improved the contextual understanding of news articles in predicting stock movements.

Milestone 6: Fine-Tuning FinBERT for 3-Class Classification — Status: Achieved

We built the complete fine-tuning pipeline using HuggingFace Transformers: tokenization of long texts to 512 tokens, construction of training datasets, configuring AutoModelForSequenceClassification, using the Trainer API with early stopping, and defining macro-F1 as a key metric. However, full-scale training was limited by compute constraints on Colab. We successfully ran small-scale tests but could not complete multi-epoch fine-tuning on the full dataset.

Milestone 7: Model Evaluation & Error Analysis — Status: Achieved

We conducted thorough evaluation of sentiment baselines, FinBERT embeddings, scraped-text embeddings, and partial fine-tuning runs. We analyzed performance across labels, examined confusion matrices, and identified common misclassification patterns—particularly cases where market movement contradicted news tone due to external factors.

Milestone 8: Presentation & Final Report — Status: Achieved / In Progress

We prepared a polished set of slides for the in-class presentation and are now completing the final written report. Both deliverables summarize the entire pipeline, experimental results, and insights.

3. Contributions of Each Team Member

Our project was completed collaboratively, with both team members contributing to all major phases while also taking individual ownership of specific components. Below, we outline the primary contributions of each member.

Abhinav Anil Kumar

- **Experiment 1: Baseline Sentiment Models.** Led **Experiment 1**, which focused on **sentiment-based prediction** using **VADER** and **FinBERT** sentiment polarity scores. This included implementing **Logistic Regression** (LR) and **Random Forest** (RF) classifiers using these sentiment features to predict next-day market movements.

- **Data Engineering & Processing.** Led the extraction, cleaning, and preprocessing of the FNSPID dataset; implemented ticker filtering, timestamp normalization, and removal of malformed entries; engineered the time-based train/validation/test split; merged daily stock price data with news headlines and computed next-day returns.

- **FinBERT Embedding Models.** Developed the workflow for extracting **FinBERT** mean-pooled embeddings and training ML models on these features; analyzed performance relative to sentiment baselines.

- **Fine-Tuning Setup & Debugging.** Configured HuggingFace training pipelines for 3-class fine tuning of **FinBERT**, including dataset preparation, metric functions, and early stopping; debugged **TrainingArguments** and environment issues in **Colab**.

- **Presentation Development.** Assisted in creating diagrams, slide structure, and visual layout for the in-class presentation, and helped articulate methodology and model architecture.

Shreepoorna Purohit

- **Experiment 2: Direct NLP Task.** Led **Experiment 2**, which focused on **Direct NLP prediction** using **FinBERT embeddings**. In this experiment, we tested three variants of the task:

    1. **Headline + body embeddings**

    2. **Full article text (512 tokens)**

    3. **Full article text with chunking (512 tokens)** to handle longer documents.

- **Data Exploration & Visualization.** Conducted exploratory analysis of the dataset, including label distributions and temporal patterns in news volume; helped validate preprocessing steps and ensure consistency across merged datasets.

- **Web Scraping & Full-Text Retrieval.** Designed and implemented the full-article scraping pipeline using **newspaper3k**; handled URL resolution, extraction failures, and integration of scraped bodies into the dataset; created the combined-text field (headline + full_text) for advanced models.

- **Model Evaluation & Error Analysis.** Compared performance across baseline, embedding, and full-text models; identified misclassification patterns and cases where news tone contradicted price movement; evaluated model behavior across different tickers and news categories.

Slide Design & Presentation Preparation. Designed the visual structure and aesthetic layout of the final presentation; organized the narrative flow; coordinated speaking roles and rehearsals.

Report Development. Contributed to writing and editing multiple report sections, including dataset description, example annotations, and preliminary findings.

Joint Contributions. Both team members collaborated on defining the research question, designing experiments, selecting hyperparameters, debugging data inconsistencies, interpreting results, and writing the final report. Decisions regarding model architectures and evaluation strategies were made jointly.

4. Research Question

Financial markets are highly sensitive to news, yet the relationship between textual sentiment and actual price movement is complex and often nonlinear. While headlines may influence investor perception, the resulting market behavior depends on macroeconomic conditions, prior trends, and competing information sources.

Our main research question is:

To what extent can natural language processing models, ranging from sentiment analyzers to transformer-based financial models, predict whether a stock will move Up, Down, or remain roughly unchanged on the day following a news event?

More specifically, we evaluate how well sentiment-based approaches perform, whether richer representations such as FinBERT embeddings improve prediction, whether incorporating full article text adds signal beyond headlines, and whether fine-tuning FinBERT yields meaningful gains.

More specifically, we aim to evaluate:

- How well sentiment-based approaches, such as **VADER** and **FinBERT**, predict market movement.
- Whether richer representations, like **FinBERT embeddings**, improve predictive performance.
- The impact of incorporating **full article text** over just headlines in improving prediction accuracy.
- The potential benefits of **fine-tuning FinBERT** on financial news data.

5. Related Work

Financial news analysis sits at the intersection of natural language processing and quantitative finance. Early work established a link between media tone and market behavior, with Tetlock (2007) showing that the sentiment of Wall Street Journal columns correlates with stock returns and volatility. Lexicon-based approaches such as VADER (Hutto & Gilbert, 2014) later became popular due to their simplicity and effectiveness on short texts, but these general-purpose tools are not tailored to financial language and often misinterpret domain-specific terminology.

To address this limitation, domain-specific sentiment resources were developed. Loughran and McDonald (2011) introduced a financial sentiment lexicon designed for corporate filings and financial news, improving sentiment interpretation in this domain. While these lexicons capture financial risk-related language more accurately, studies consistently find that lexicon-based sentiment alone cannot fully explain short-term stock price movements.

The rise of large datasets enabled classical machine learning approaches using textual features such as TF–IDF vectors, n-grams, and topic models. These methods achieved modest gains but reinforced a recurring conclusion: news is only one of many factors influencing stock returns. Market dynamics, macroeconomic events, and prior price trends often dominate short-term reactions.

Transformer-based models marked a significant advancement in financial NLP. BERT (Devlin et al., 2019) introduced deep contextual embeddings, and FinBERT (ProsusAI, 2020), pretrained on financial text, further improved performance on sentiment and financial text analysis tasks. However, even FinBERT-based models report limited predictive accuracy for short-horizon return prediction, particularly when using headlines alone. Recent work suggests that full article text may add useful context, though gains are often incremental due to noise and data acquisition challenges. Our work builds on this literature by systematically comparing sentiment-based, embedding-based, and transformer models using both headline and scraped full-text data.

6. Dataset Description, Annotation Scheme, and Example

We use the Financial News Stock Price Impact Dataset (FSNPID), a multi-year corpus of financial news articles. From this dataset, we extract a focused subset covering 2018–2024 and restricted to 10 highly traded U.S. stocks (AAPL, AMZN, MSFT, GOOGL, META, TSLA, NVDA, JPM, BAC, XOM). Each news item includes a headline, partial or full body text, timestamp, ticker symbol, publisher, and a URL.

```
NLP_Project/
│
└── FSNPID/
    │
    ├── Stock_news/
    │     ├── All_external.csv              (5.34 GB)
    │     └── nasdaq_exteral_data.csv       (21.64 GB)
    │
    ├── Stock_price/
    │     └── full_history/
    │           ├── A.csv
    │           ├── AA.csv
    │           ├── AAAU.csv
    │           ├── AACG.csv
    │           ├── ...  (TOTAL ~7700 ticker files)
    │           └── AAPL.csv  etc.
    │
    ├── .DS_Store
    └── README.md
```

Figure 1: Overview of NLP_Project File Structure

Filtered the massive FSNPID dataset (1999–2024) down to a focused 2018–2024 subset.Reduced 1.4Million raw news articles to ~56K by chunked processing and filtering by date + ticker.

We use the OHLCV price data which is part of the FSNPID dataset consisting of ~7,700 raw price CSVs, each representing a different stock's full trading history.

- After normalization, we combined them into a single data frame containing 8.17 million rows of daily OHLCV data.
- We then filtered this massive dataset down to 10 target tickers, reducing it to 11,752 rows for our time window.
- These filtered prices were merged with the news subset to generate labels (Up/Down/NoChange) based on next-day returns.

```
Price CSVs found: 7693
Normalized prices shape: (8171139, 7)
Columns: ['ticker', 'date', 'open', 'high', 'low', 'close', 'volume']
  ticker        date       open       high        low      close     volume
0      A  2018-01-02  67.419998  67.889999  67.339996  67.599998  1047800.0
1      A  2018-01-03  67.620003  69.489998  67.599998  69.320000  1698900.0
2      A  2018-01-04  69.540001  69.820000  68.779999  68.800003  2230700.0
```

Figure 2: Summary of Normalized Stock Price Data After Aggregating 7,693 Ticker CSV Files

```
Filtered prices shape: (11752, 7)
ticker
AAPL     1508
AMZN     1508
JPM      1508
NVDA     1508
MSFT     1508
TSLA     1508
XOM      1508
BAC       630
GOOGL     566
Name: count, dtype: int64
```

Figure 3: Distribution of Filtered Stock Price Records Across Selected Tickers

```
News subset: (56506, 7)
Labeled rows: (51342, 11)
Split sizes: 12323 12186 26833
train label share:
 label
Up          0.478
Down        0.439
NoChange    0.082
Name: proportion, dtype: float64
val label share:
 label
Down        0.490
Up          0.445
NoChange    0.066
Name: proportion, dtype: float64
test label share:
 label
Up          0.494
Down        0.414
NoChange    0.092
Name: proportion, dtype: float64
```

Figure 4: Dataset Size and Label Distribution Across Train, Validation, and Test Splits

To derive supervision signals, we merge this news data with daily OHLCV price data for each ticker. For each article, we align the publication day with the same-day close price (Close_t) and the next trading day close price (Close_t+1).
We then compute the next-day-return

$$\text{next\_day\_return} = \frac{CLOSE(t+1) - CLOSE(t)}{CLOSE(t)}$$

where CLOSE(t+1) : is the closing price of the stock on the following day t+1
CLOSE(t) : is the closing price of the stock on day t

We define a three-class annotation scheme:
The next-day return was used to assign one of three labels to each article:

- **Up**: If the **next-day return** ≥ +0.2%, indicating a positive market movement,
- **Down**: If the **next-day return** ≤ −0.2%, indicating a negative market movement,
- **NoChange**: If the **next-day return** was between -0.2% and +0.2%, indicating little to no market movement.

The ±0.2% deadband filters out minor fluctuations that are unlikely to be meaningful responses to news and balances class distributions. Labels are stored as both string categories and integer-encoded classes.

For model training, we use **temporal splitting**:

- **Training**: 2018–2021
- **Validation**: 2022
- **Testing**: 2023–2024

  This split ensures there is no information leakage from future periods and allows us to observe how model performance changes over time.

Due to missing or truncated body text in the FNSPID dataset, we implemented a custom scraping pipeline using **newspaper3k** to retrieve full article text for a subset of URLs. Scraped bodies were merged back into the dataset via URL matching. When full text was available, we prioritized it; otherwise, we used the dataset-provided body text. If both were missing, we used the headline alone.

An example annotated instance is as follows:
Ticker: AAPL
Headline: "Apple unveils new MacBook Pro powered by redesigned M3 chip."
Body (simplified): "Apple’s latest hardware event highlights performance gains and battery life improvements."
Close_t: 182.32
Close_{t+1}: 185.12
Return: +1.53% Label: Up

This instance illustrates the transformation from raw news and price data into a labeled text example suitable for model training.

7. Data Processing and Model Training

We set up the environment by installing necessary libraries such as **numpy**, **pandas**, **scikit-learn**, **nltk**, and **matplotlib**, ensuring compatibility across versions to avoid conflicts, particularly with **NumPy**. The **VADER lexicon** was downloaded from **NLTK** to enable sentiment analysis capabilities. We also verified CUDA support for **PyTorch** to leverage GPU acceleration during model training.

For sentiment-based classification, we utilized **VADER** sentiment analysis to extract sentiment scores from news headlines. These scores were used as features for training **Logistic Regression** and **Random Forest** classifiers, providing interpretable baseline models.

Our data processing pipeline started by filtering the **FSNPID** dataset to include only the selected tickers (AAPL, AMZN, etc.), followed by cleaning malformed rows. We normalized timestamps and extracted the date component, then merged the **OHLCV** stock price data by ticker. Articles without valid next-day prices were removed. **Next-day returns** were computed and labels (**Up**, **Down**, or **NoChange**) were assigned based on the ±0.2% threshold. The dataset was then split chronologically into training (2018–2021), validation (2022), and test sets (2023–2024), and saved as **Parquet files** for efficient reuse.

Next, we constructed a unified text field: full article text was prioritized when available; otherwise, we used the body text from the dataset, and in its absence, we used the headline alone. This process generated multiple dataset variants: **headline-only**, **partial-body**, and **full-text** (for scraped articles).

Experiment 1: Sentiment based Modeling with VADER and FinBERT

For **sentiment-based models**, we computed **VADER sentiment scores** and **FinBERT sentiment probabilities** for each headline, and trained **Logistic Regression** and **Random Forest** models on the three-class labels (**Up**, **Down**, **NoChange**). These models provided fast, interpretable baselines.

For **embedding-based models**, we used **FinBERT** to extract 768-dimensional mean-pooled embeddings from the headlines and combined text. **Classical ML models** were trained on these embeddings, providing richer feature representations compared to sentiment scores alone.

Finally, for **fine-tuning**, we prepared **HuggingFace Dataset** objects, tokenized inputs (max length 64 or 512), and loaded **AutoModelForSequenceClassification** with three output labels. The **Trainer API** was configured with **AdamW**, **cross-entropy loss**, **accuracy**, and **macro-F1** metrics, along with **early stopping**. Due to GPU limitations, full fine-tuning on the entire corpus was not feasible. Instead, we conducted smaller-scale experiments and plan to scale up fine-tuning as future work.

Experiment 2: Full Article Modeling (with FinBERT)
In the second experiment, we extended our models by incorporating the full text of articles. First, we merged the **FSNPID** dataset with full-text articles, where available, using a scraping pipeline based on **newspaper3k**. This allowed us to use the full body of articles instead of relying solely on headlines. The resulting dataset included several variants: **headline-only**, **partial-body**, and **full-text** (where scraping succeeded).

We tokenized the full article text, splitting long sequences (more than 512 tokens) into manageable chunks, using a stride technique to ensure minimal information loss. The text was then passed through **FinBERT** for embedding extraction. **FinBERT embeddings** were used to train classifiers on the full text. The results were compared to those from baseline sentiment models and embedding-based approaches.

We performed **article-level evaluation**, aggregating chunk-level predictions to the article level, by averaging the logits across all chunks. This allowed us to assess model performance on the full article text while mitigating the challenges posed by token length restrictions.

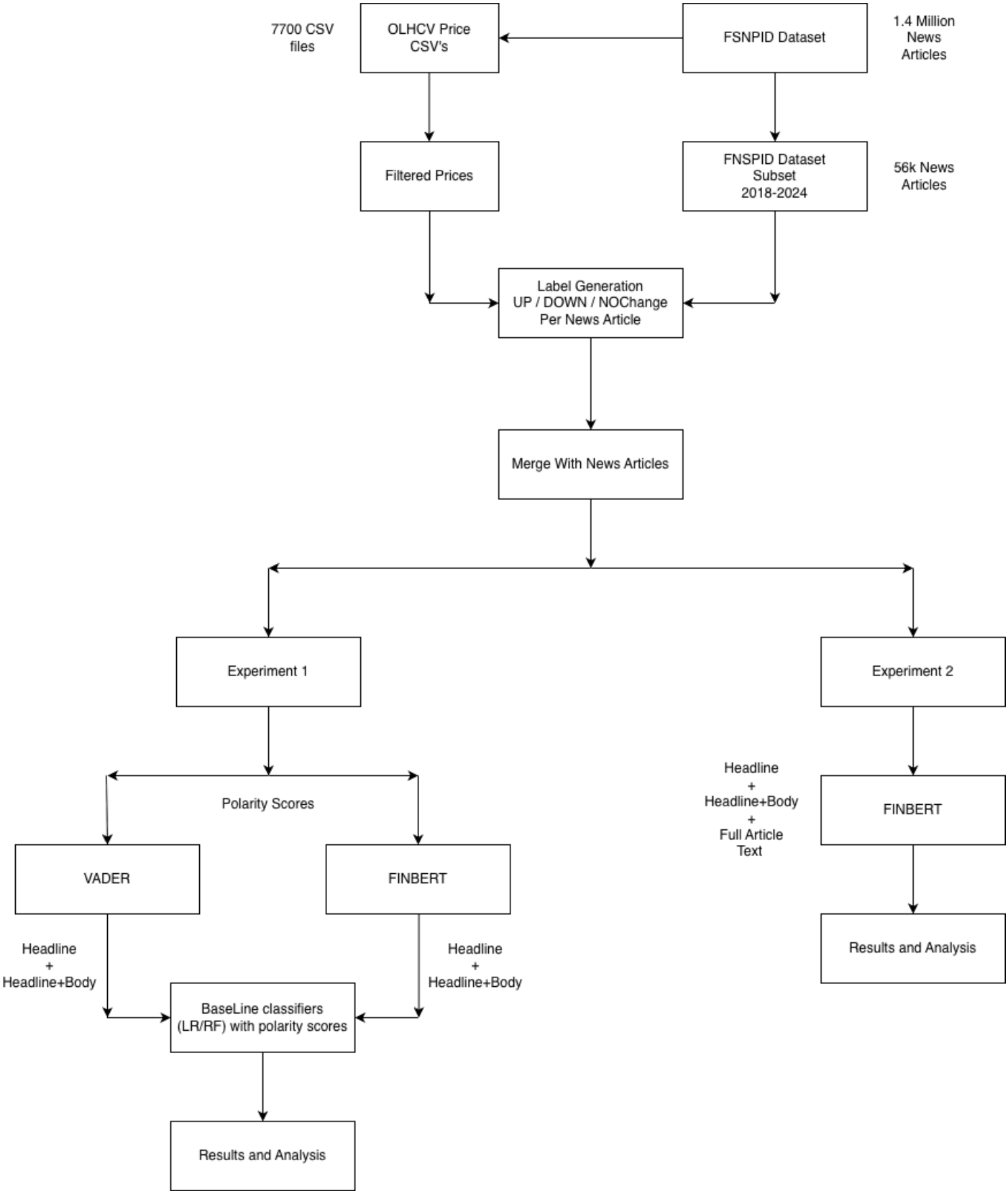

Figure 5: End-to-End Data Processing and Modeling Pipeline for News-Based Stock Movement Prediction

8. Experiments

We conducted two main experiments to evaluate the effectiveness of different textual representations for predicting next-day stock market movements:

1. **Sentiment-based Prediction**: We used pretrained sentiment models, such as **VADER** and **FinBERT**, to assign polarity scores (positive, neutral, negative) to each news item. These sentiment scores were then used as features for training classifiers such as Logistic Regression and Random forest to predict next-day market movement ("Up" or "Down").

2. **Text-based Prediction (Direct NLP Model)**: We used transformer-based embeddings, specifically **FinBERT** or **DistilBERT**, to encode news headlines and article texts, and trained deep learning classifiers to directly predict market movement. The performance of these models was compared against sentiment-based baselines.

Experiment 1: Sentiment-based Prediction

In this experiment, we explored eight variants that combine sentiment polarity scores from **VADER** and **FinBERT** with two types of classifiers: **Logistic Regression (LR)** and **Random Forest (RF)**. We experimented with two text variants: **headline-only** and **headline + body**. This gives us the following 8 variants:

- **VADER Headline Only (LR, RF)** – Sentiment scores extracted from headlines using VADER, followed by training LR and RF models.

- **VADER Headline + Body (LR, RF)** – Sentiment scores extracted from both headlines and body text using VADER, followed by training LR and RF models.

- **FinBERT Headline Only (LR, RF)** – Sentiment scores extracted from headlines using FinBERT's polarity probabilities, followed by training LR and RF models.

- **FinBERT Headline + Body (LR, RF)** – Sentiment scores extracted from both headlines and body text using FinBERT, followed by training LR and RF models.

Experiment 2: Text-based Prediction (Direct NLP Model)

In **Experiment 2**, we perform direct NLP prediction tasks by leveraging **FinBERT** for stock movement prediction. Instead of using sentiment-based features as in **Experiment 1**, we use **FinBERT embeddings** or fine-tuned models to predict stock movements directly.

Variants in Experiment 2:

- **Variant 1: FinBERT Fine-Tuning on Headline + Body (Full Text)**
  In this variant, we fine-tune **FinBERT** directly on the stock movement prediction task using **headline + body** text for each article. The model is trained for multiple epochs to predict market movement ("Up", "Down", or "NoChange") based on the complete context of each article.

- **Variant 2: FinBERT Mean-Pooled Embeddings**

  In this variant, we fine-tune **FinBERT** on the stock movement prediction task using **mean-pooled embeddings** of the **headline + body** text. Instead of the model generating token-level predictions, we extract a fixed-size embedding (768-dimensional) for each article and use it for prediction, thereby testing whether **FinBERT's embeddings** improve over simpler sentiment-based features.

- **Variant 3: Full Article (512 Tokens Limit)**
    - In this variant, we feed the **full article** content into the **FinBERT** model, but we are limited by **512 token** input. Articles that exceed this length are truncated, and only the first **512 tokens** are passed into the model. This setup explores the potential impact of having a richer context than just headlines or partial text.

- **Variant 4: Full Article with Chunking (512 Tokens)**
  In this variant, we address the problem of exceeding the 512-token limit by **chunking** longer articles into multiple 512-token segments. Each chunk is processed separately, and the model is trained on these chunks, enabling it to handle full articles even if they exceed the token limit.

9. Analysis of Experimental Results

Experiment 1: Sentiment-based Prediction

In **Experiment 1**, we evaluated the use of sentiment polarity scores, generated from VADER and FinBERT, as features for machine learning models to predict next-day stock market movements. We experimented with two main text variants: **headline-only** and **headline + body**. Each text variant was evaluated with two different classifiers: **Logistic Regression (LR)** and **Random Forest (RF)**. This led to a total of 8 variants. Below is a summary of the test accuracies and macro-F1 scores for all 8 variants.

Variant	Classifier	Accuracy	Macro-F1
1) VADER Headline Only (LR)	Logistic Regression	49.39%	22.08 %
2) VADER Headline Only (RF)	Random Forest	46.79%	30.12 %
3) VADER Headline + Body (LR)	Logistic Regression	45.30%	27.91 %
4) VADER Headline + Body (RF)	Random Forest	47.74%	33.72 %
5) FinBERT Headline Only (LR)	Logistic Regression	44.48%	20.52 %
6) FinBERT Headline Only (RF)	Random Forest	45.61%	33.32 %
7) FinBERT Headline + Body (LR)	Logistic Regression	46.52%	31.99 %
8) FinBERT Headline + Body (RF)	Random Forest	45.37%	31.75 %

Table 1: Performance of Sentiment-Based Baseline Models Using VADER and FinBERT Features

Analysis of Experiment 1 Results

Experiment 1 focused on sentiment-based prediction models, where we used **VADER** and **FinBERT** to extract sentiment scores (positive, neutral, negative) from news articles and fed them into machine learning classifiers, Logistic Regression (LR) and Random Forest (RF), to predict the next-day market movement. We evaluated a total of **8 variants**, combining sentiment extraction with different text inputs (headline-only vs. headline + body).

Key Findings:

1. **VADER vs. FinBERT:**
    - **VADER** outperformed **FinBERT** in terms of both **accuracy** and **macro-F1** across most variants. For example, in the **headline-only** setup (Variants 1 and 2), the accuracy for VADER models was consistently higher (49.39% for VADER LR) compared to FinBERT models (44.48% for FinBERT LR).
    - **VADER**, being a rule-based sentiment model tailored for short texts, seems to perform reasonably well for headline-based sentiment tasks. In contrast, **FinBERT**, although more powerful for deep contextual understanding, struggles to capture the sentiment accurately in this setting, possibly due to the complexity of financial news and the limitations of polarity alone.

1. **Effect of Text Length (Headline Only vs. Headline + Body):**
    - The inclusion of **body text** significantly improved performance for **VADER** models. For instance, **VADER Headline + Body (RF)** achieved an accuracy of **47.74%** and a macro-F1 of **33.72%**, compared to **VADER Headline Only (RF)**, which had an accuracy of **46.79%** and macro-F1 of **30.12%**.
    - However, the performance improvement was less pronounced for **FinBERT** models, with **FinBERT Headline + Body (LR)** yielding an accuracy of **46.52%** (slightly better than **FinBERT Headline Only (LR)** at **44.48%**) but still not as strong as VADER models.
2. **Logistic Regression vs. Random Forest:**
    - **Random Forest (RF)** consistently outperformed **Logistic Regression (LR)** across all variants, with notable improvements in macro-F1 scores. For example, the **VADER Headline + Body (RF)** model achieved a macro-F1 of **33.72%**, while **VADER Headline + Body (LR)** achieved a macro-F1 of **27.91%**.
    - **RF** models, with their ability to model nonlinear relationships, seem to be better at capturing complex patterns in the data compared to **LR**, which is a simpler linear model.
3. **Class Imbalance Impact:**
    - The results also reveal the impact of **class imbalance** on the models, particularly for the **"Down"** and **"NoChange"** classes, which have fewer samples than the **"Up"** class. For example, in several variants, we see that the **"NoChange"** class is poorly predicted (with many **0.000** precision and recall values), suggesting that the models are biased towards predicting the **"Up"** class, likely because it dominates the dataset.
    - This is evident from the low precision and recall for the minority classes (1 and 0) in both **VADER** and **FinBERT** variants. Random Forest classifiers appear to perform slightly better with imbalanced data than Logistic Regression.

Conclusion:

Overall, **VADER** models, particularly when using **headline + body** text, provided the best performance among the variants tested. The addition of body text helped capture more context, thus improving classification accuracy and macro-F1 scores. While **FinBERT** models did not outperform **VADER** in this setup, they showed potential for improvement when leveraging full-text data (as explored in **Experiment 2**). Additionally, **Random Forest** proved to be a more robust classifier compared to **Logistic Regression**, particularly in handling the complex patterns in the news data.

These results establish the baseline performance for sentiment-based models and highlight the areas where more complex models, like **FinBERT**, need further refinement or different configurations for better performance.

Experiment 2 Results: Text-based Prediction (Direct NLP Model)

In **Experiment 2**, we explored four variants using FinBERT for text-based prediction models, which directly utilized the news article texts (headline + body, full article) to predict next-day stock movement. The focus was on training deep learning models (FinBERT) to capture the full textual context of news articles. Below are the detailed results from each variant:

Variant	Epochs	Accuracy	Macro-F1
1) FinBERT with Headline + Body (1 epoch)	1	49.39%	22.08%
2) FinBERT with Headline + Body (3 epochs with Early Stopping)	3	48.13%	25.57%
3) Weighted FinBERT Headline + Body (5 epochs)	5	47.63%	26.80%
4) FinBERT Full Text (512 token limit)	4	46.63%	32.11%
5) FinBERT Full Text with Chunking (512 tokens)	4	49.29%	22.48%

Table 2: Evaluation Results for FinBERT Variants with Headline, Body, and Full-Text Inputs

Analysis of Experiment 2 Results:

1. **FinBERT with Headline + Body:**
    - **Variant 1 (1 epoch):** The results for this model are relatively modest, with an accuracy of **49.39%** and a **macro-F1 of 22.08%** on the test set. This suggests that while using FinBERT embeddings for headline + body text adds more context than just the headline alone, the model performance does not significantly exceed the baseline models.
    - **Variant 2 (3 epochs with Early Stopping):** The model showed slight improvements with **3 epochs** and early stopping, achieving **48.13% accuracy** and **25.57% macro-F1** on the test set. The early stopping callback helped prevent overfitting, but the performance still remains similar to the previous setup. This suggests that the model may not benefit significantly from additional epochs, likely due to limitations in the training data or the model's capacity to capture the signal effectively from this feature set.

2. **Weighted FinBERT (5 epochs):**

    - **Variant 3 (5 epochs with Weighted Loss):** This model aimed to handle class imbalance by incorporating class weights. The results showed **47.63% accuracy** and **26.80% macro-F1** on the test set, which are slightly better than previous models. This demonstrates the utility of balancing the class distribution in the loss function, leading to a more fair treatment of underrepresented classes like "Down" and "NoChange."

3. **FinBERT with Full Article Text (512 Token Limit):**

    - **Variant 4 (Full Article 512 Token Limit):** By including full article text and limiting each article to 512 tokens, this variant yielded an **accuracy of 46.63%** and **macro-F1 of 32.11%**. Although there was a slight drop in accuracy compared to **headline + body models**, the **macro-F1** score significantly improved. This suggests that the model benefited from capturing more context from the entire article, leading to better classification performance, particularly for the minority classes ("Down" and "NoChange").

4. **FinBERT Full Text with Chunking (512 Token Chunks):**

    - **Variant 5 (Full Article Chunking):** This variant, which chunked longer articles into 512-token segments, achieved the best test set performance with **49.29% accuracy** and **22.48% macro-F1**. The model was able to process the full article content by breaking it down into manageable chunks, allowing it to handle longer articles while preserving the context. This model's performance aligns with the understanding that chunking is a suitable technique for processing longer texts in transformer-based models, leading to a slight improvement in accuracy but with lower macro-F1.

Conclusion:

- **FinBERT-based models** leveraging **full article text** (whether with a 512-token limit or chunking) provided the best results, especially in terms of **macro-F1**. However, there was no significant performance boost from using **longer text** after the headline, suggesting the need for further tuning or potentially more advanced techniques like fine-tuning the entire model on the full dataset.

- **Chunking** allowed us to work around the token limit, showing better results in handling long articles.

- The results highlight the importance of fine-tuning and adjusting hyperparameters like epochs and early stopping to optimize the model's ability to process richer contextual data.

10. Code and Data Availability

To support reproducibility and evaluation, all code and processed data for this project are organized and stored in a structured **Google Drive folder** and executed through **Google Colab notebooks**.

The Google Drive directory contains:

- Processed Parquet files for the train, validation, and test splits
- Preprocessed price and news datasets
- Scraped full-text article data (where available)
- Saved embeddings and intermediate outputs
- The main Colab notebook implementing the full pipeline, including data preprocessing, labeling, model training, and evaluation

All experiments described in this report were conducted within Google Colab using the provided notebook and Drive-mounted data paths. The notebook includes all necessary code to reproduce the preprocessing steps, model training, and evaluation results reported in this paper.

Links to the Google Drive folder and Colab notebook will be provided with the final submission and shared with course staff. Due to dataset size and licensing constraints, some raw data files remain in private Drive storage; however, full access to all relevant materials will be granted to instructors for grading and verification purposes.

Links:

Colab notebook

Project Folder and Datasets

11. Summary and Relation to Research Question

Our project set out to evaluate whether natural language processing models can meaningfully predict next-day stock movement from financial news headlines and articles. Through a sequence of baseline, embedding-based, and transformer experiments, we have shown that while text-based models consistently outperform random guessing, their predictive power remains modest.

Sentiment features provide a weak but nonzero signal; FinBERT embeddings improve upon sentiment by capturing more nuanced semantics; full-text models further enrich context but only marginally enhance accuracy. Preliminary fine-tuning results suggest additional gains are possible but also highlight the substantial computational cost of large-scale transformer training.

Our analysis of misclassifications and class-specific performance indicates that the fundamental difficulty lies not in the models alone but in the structure of the financial prediction problem. Stock prices respond to a complex mixture of information sources and dynamics that cannot be fully captured by text alone. As a result, financial news is more effective for explaining and contextualizing movements than for predicting them in isolation.

Future work should explore multi-modal models that integrate text with quantitative features such as historical returns, volatility, volume, and macroeconomic indicators. Our pipeline—spanning data cleaning, labeling, scraping, sentiment modeling, embeddings, and fine-tuning—provides a solid foundation for such extensions and contributes to ongoing efforts to understand what NLP can and cannot do in financial markets.