Real-Time Sentiment-Aware News Ticker for Financial Markets

Toka Nabil[1], Sama Zaky[1], Sara Yosry Mohamed[1], Rana Medhat Elsayed[1]
{T.Nabil2189, s.Ayman2176, s.yosry2288, R.medhat2241}@nu.edu.eg
CS program, School of Information Technology and Computer Science (ITCS), Nile University, Giza, Egypt

The financial market is increasingly influenced by public sentiment expressed across social platforms and news media. This paper presents a real-time sentiment-aware stock movement prediction system that integrates streaming data from Reddit and financial news APIs into a structured PostgreSQL data warehouse. Unlike static models, our pipeline supports continuous training using new data every twenty minutes, and updates predictions through a XGBoost classifier, achieving an accuracy of up to 72%. The model incorporates engineered features such as sentiment scores, time categories, and sentiment strength. Visualization through Power BI Dashboard enables immediate decision-making. Our system demonstrates reliable ingestion and processing times—averaging 0.8 to 1.3 seconds per news cycle and 2.8 to 3.4 seconds for Reddit posts—making it suitable for live trading support. This framework bridges the gap between sentiment extraction and real-time predictive modeling in financial applications.

***Index Terms*—Sentiment Analysis, Stock Market, Reddit API, News API, Real Time, yfinance**

I. Introduction

The stock market is a highly reactive environment shaped by a multitude of factors including economic indicators, investor behavior, and notably, public sentiment. Traditional stock prediction methods rely heavily on historical price patterns and technical indicators, often failing to incorporate the growing influence of online discourse. Platforms such as Reddit and financial news outlets have become essential in shaping investor perception and thus market trends.

Sentiment analysis provides a means to quantify and leverage this unstructured textual data. Prior research has demonstrated the effectiveness of sentiment-enhanced models in stock prediction. For instance, logistic regression, random forests, and XGBoost have been applied to sentiment-enriched datasets with promising outcomes [7], [8], [22]. FinBERT and other transformer-based models have further advanced sentiment classification accuracy in financial contexts [4], [5], [6]. However, most of these studies are constrained to static datasets and offline models. Real-world applications demand models that not only capture sentiment from multiple platforms but also adapt in real time. Moreover, processing latency, data heterogeneity, and infrastructure limitations are often overlooked in academic setups [20], [21].

Our research addresses these challenges by implementing a live system that streams and processes sentiment data from Reddit and NewsAPI, stores it in a PostgreSQL data warehouse then transfers it to Power BI for data visualization, and trains an XGBoost model at twenty-minute intervals. The classifier uses features such as ticker, sentiment score and sentiment time category to predict stock movement with an accuracy reaching 72 percent. The ingestion pipeline ensures data fetch, preprocessing, and insertion within 0.5–3 seconds on average, enabling seamless model retraining. This architecture delivers a practical, high-performance solution for integrating sentiment into real-time financial forecasting.

II. Related Work

Sentiment analysis has become a critical component in stock market prediction, complementing traditional financial indicators. Numerous studies have demonstrated the potential of integrating social media and news sentiment with machine learning techniques to forecast stock price movements.

Bollen et al. [1] explored how aggregate mood levels extracted from Twitter can predict broad market trends. Similarly, Zhang et al. [2] and Nguyen et al. [3] analyzed Twitter data to forecast market indicators and demonstrated that mood indicators precede financial changes. These early efforts established the viability of using online sentiment for financial modeling.

Recent advancements have focused on combining sentiment signals with advanced classifiers. Araci [4] introduced FinBERT, a domain-specific BERT model fine-tuned for financial sentiment classification, significantly improving accuracy over generic models. Jiang and Zeng [5], as well as Adelakun and Adebisi [6], applied FinBERT in predicting stock movement using financial news, highlighting the utility of transformer-based architectures.

Machine learning approaches including Random Forests, XGBoost, and LightGBM have also been extensively applied to stock prediction tasks. Studies such as those by Basak et al. [7], Khaidem et al. [8], and Daori et al. [9] showed strong performance using tree-based classifiers. Vijh et al. [10] and Werawithayaset and Tritilanunt [11] further validated these models by comparing them with traditional regression-based methods.

Hybrid models integrating numerical and textual inputs have also gained traction. Akita et al. [12] utilized deep learning to combine financial metrics with sentiment. Karim and Ahmed [13] employed BiGRU and BiLSTM for improved sequence modeling, while Maqbool et al. [14] used MLP regressors for sentiment-enhanced prediction. Duong et al. [15] and Sariyer et al. [16] focused on financial news from specific stock exchanges, such as Ho Chi Minh and BIST30, respectively.

The news domain has also proven essential. Li et al. [17] analyzed how news content influences stock returns using sentiment scores. Teti et al. [18] explored the correlation between Twitter activity and tech stock movements, revealing a strong association. Eachempati et al. [19] highlighted sentiment shifts during the COVID-19 pandemic and their effects on stock markets.

Pagolu et al. [20] and Adlakha et al. [21] contributed real-time analysis frameworks that incorporate Twitter sentiment for live market predictions, emphasizing the importance of latency and system design in financial applications.

While previous studies have demonstrated the potential of sentiment analysis in financial forecasting, they often rely on static datasets, offline models, or single data sources such as Twitter or news. Real-time systems that integrate multi-source sentiment streams, synchronize them with live stock prices, and retrain predictive models dynamically are largely absent. Moreover, the effect of sentiment timing—whether expressed before, during, or after market hours—remains underexplored. Our study addresses these gaps by developing a real-time sentiment-aware forecasting system that streams Reddit and news sentiment, matches it with live yfinance stock data, classifies price movement and time-of-day categories, retrains models every 20 minutes, and visualizes results in a dynamic Power BI dashboard.

III. Methodology

A. System Architecture

In our study, we developed a comprehensive methodology to build a real-time sentiment-aware news ticker for financial markets. Our pipeline comprises several distinct stages as shown in figure blabla We collected real-time data from Reddit and News APIs and retrieved stock prices using the yfinance library, performed preprocessing to clean and standardize the data, integrated and stored the results in a PostgreSQL database, conducted data modeling and dashboard visualization using Power BI, applied machine learning models(Bert, XGBoost, Ensemble(Random forest, Logistic regression and XGBoost)) for stock movement prediction, used a RAG model for market analysis, and deployed the system using FastAPI with Mistral AI for the backend and Streamlit for the user interface.

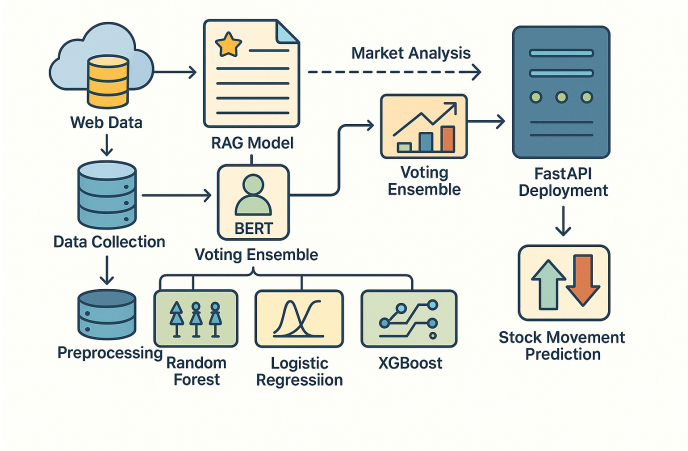

Fig. 1. System Architecture

B. Data collection

To build a system capable of tracking public sentiment on financial topics, we collected data from three primary sources. First, we employed the Python praw library to stream Reddit posts in real-time from finance-centric subreddits such as r/wallstreetbets and r/stocks, it fetches all the new posts each five? minutes. These posts were filtered to capture those containing references to stocks or financial keywords. From each post, we extracted the title, textual content, author, and timestamp. Second, we retrieved financial news articles using the NewsAPI, it fetches all the new posts each 20? minutes, filtered by stock-related keywords to ensure relevance. Extracted metadata included the article's title, content, source name, and publication timestamp. These news articles complement social media data by offering a formal and editorial perspective on market events. Finally, to acquire real-world market data, we used the yfinance library to download daily historical price data for publicly traded companies, it fetches all the new data each day. For each ticker symbol, we retrieved the open, close, high, and low prices, as well as trading volume. This data serves as the ground truth for evaluating the accuracy of sentiment-based predictions. We collected data from May 3 to May 31, 2025, resulting in a total of 19,029 entries, including sentiment from Reddit and news, along with stock prices.

C. Data Pre-processing

After data collection, we conducted a series of preprocessing steps to standardize and prepare the data for analysis. All text entries were cleaned to remove URLs, emojis, special characters, and extraneous whitespace. This normalization process ensured consistency across both Reddit and news content. A company-ticker keyword map was used to extract stock mentions from the cleaned text. This step allowed us to link each sentiment entry to one or more specific ticker symbols. Sentiment scores were then computed using the VADER (Valence Aware Dictionary and sEntiment Reasoner) sentiment analysis tool. The analyzer returned a compound sentiment score between -1 (strongly negative) and +1 (strongly positive). Each score was mapped to a prediction label: "Likely Up," "Likely Down," or "Neutral," depending on its magnitude and polarity.

Original Text:

Wait, what? $DUOL pricing Am I missing something? Why in the world is $DUOL priced at $24B? Its trading at 255 P/E and 26 P/B. Are markets broken? Haha. Makes no sense for an app that notoriously sucks at actually teaching you other languages

Fig. 2. Text before pre-processing

Cleaned Text:

wait what $duol pricing am i missing something why in the world is $duol priced at $24b its trading at 255 pe and 26 pb are markets broken haha makes no sense for an app that notoriously sucks at actually teaching you other languages

Fig. 3. Text after pre-processing

D. Data Integration

After preprocessing, we performed a data integration step to combine sentiment data with market performance data. The sentiment entries from the fact sentiment table were merged with stock price data from the yfinance table using shared ‘ticker‘ and ‘price_date‘ fields. This process produced a unified view that aligned each sentiment prediction with its corresponding market behavior for that day. To evaluate how sentiment related to actual stock movement, we derived a new column called ‘movement‘. If the closing price (‘close‘) exceeded the opening price (‘open‘), the stock was marked as having moved Üp¨; otherwise, it was marked as Ddown.T̈his final integrated table served as the foundation for further modeling, visualization, and performance evaluation.

E. Data Warehousing and Modeling

To support scalable and efficient analysis of sentiment and financial data, we initially stored all collected and preprocessed data in a PostgreSQL database. Once cleaned and processed, the data was organized into two main tables: one for sentiment data retrieved from Reddit and news APIs and another for financial price data collected from yfinance. The first table—referred to as the fact sentiment table—included fields such as sentiment ID, ticker symbol, sentiment score, and sentiment prediction. The second table, based on yfinance data, contained daily stock prices with columns such as ticker, price date, open, and close. Once the data was structured in PostgreSQL, it was transferred to Microsoft Power BI for the purpose of advanced data modeling and visualization. To support scalable and efficient analysis of sentiment and financial data, we implemented a snowflake schema architecture within our PowerBI data warehouse. This schema design organizes the data into normalized dimension tables and fact tables, reducing redundancy and enabling high-performance queries. As shown in (fig 4) at the core of the schema lies the fact sentiment table, which stores the main sentiment analysis results from both Reddit and news sources. Each row represents a unique sentiment entry, identified by a sentiment id. This table contains the cleaned text(content), sentiment score, prediction label and foreign keys referencing the dim author and dim time tables. To represent cases where a single text references multiple companies, we introduced a bridge table called fact sentiment ticker. This table captures the many-to-many relationship between sentiment entries and ticker symbols by linking sentiment id with ticker id. Supporting dimension tables include dim ticker, which stores metadata about stock ticker symbols; dim author, which contains information about Reddit users or news sources; and dim time, which decomposes each timestamp into granular units such as day, month, and hour to facilitate temporal analysis. A separate stock prices table was maintained for historical market data. This table, derived from the yfinance API, includes daily open, close, high, low, and volume data for each ticker. Though not normalized like the sentiment tables, this structure allows for effective comparison between predicted sentiment and real-world price movements. This snowflake schema enhances the flexibility and integrity of our data model. It supports modular updates to dimension data, efficient joining of sentiment and market data, and seamless integration with visualization platforms such as Power BI.

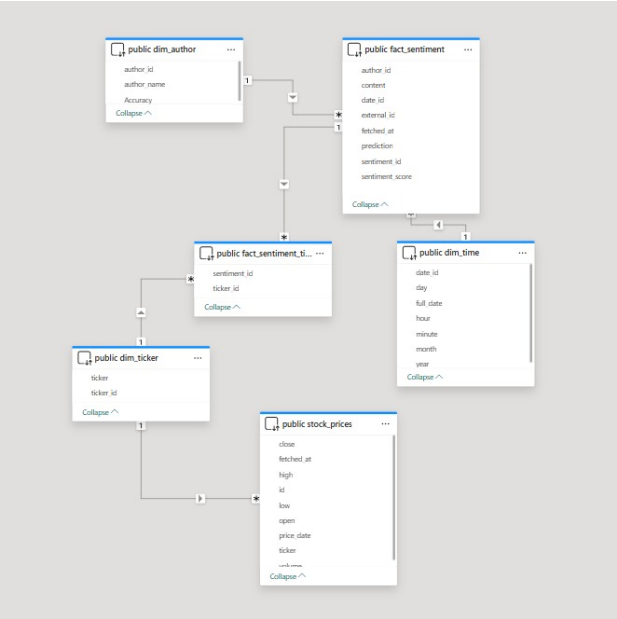

Fig. 4. Snow Flake Schema

F. Machine Learning Models

To further analyze the relationship between sentiment and stock performance, we developed a series of machine learning models aimed at two classification tasks: predicting the directional movement of a stock’s price (”Up” or ”Down”) and identifying the time category in which the sentiment event occurred (i.e., before market hours, during market hours, or after market hours). The models were trained using the unified dataset containing ticker symbols, sentiment predictions, and timestamps. We engineered features including the sentiment score and ticker name. Additionally, the sentiment time category was categorized into three discrete time classes: ”Pre-market” (before 9:30 AM), ”Market hours” (9:30 AM to 4:00 PM), and ”After hours” (after 4:00 PM), using Eastern Time conventions. We employed several models in our experiments, including Random Forest, BERT-based classifier, LSTM, XGBoost, and LightGBM (LGBM). Among these, we observed that the distribution of the target variable for price movement was imbalanced, with approximately 12,000 ”Up” instances and 4,000 ”Down” instances. To mitigate this, we applied random undersampling to the majority class when training the BERT and XGBoost models, improving the balance and generalization performance.

All models were trained and updated in near real-time, with retraining occurring every 20 minutes on the complete set of data collected up to that point. This approach ensured that

the models adapted dynamically to new sentiment inputs and market conditions.

G. Integrative of fastapi, mistral ai and streamlit for interactive prediction

To operationalize our predictive system and deliver actionable insights to end users, we developed an intelligent financial assistant by integrating FastAPI, Mistral AI, and Streamlit into a cohesive prediction pipeline. At the core, we trained a BERT-based binary classification model to predict stock movement (up or down) using financial sentiment data stored in a PostgreSQL database. This model was served using FastAPI, which exposes a RESTful /predict endpoint that accepts stock tickers and returns model predictions in real time. To enhance the interpretability of the results, we connected this backend to Mistral AI, a language model API, which rephrases raw predictions into natural, user-friendly explanations. Finally, we built a sleek user interface using Streamlit as shown in fig. lala, allowing users to input stock tickers through a simple web form and receive both the prediction and the AI-generated explanation instantly. This end-to-end pipeline demonstrates how real-time machine learning, language models, and modern web frameworks can be combined to deliver interactive and intelligent financial insights.

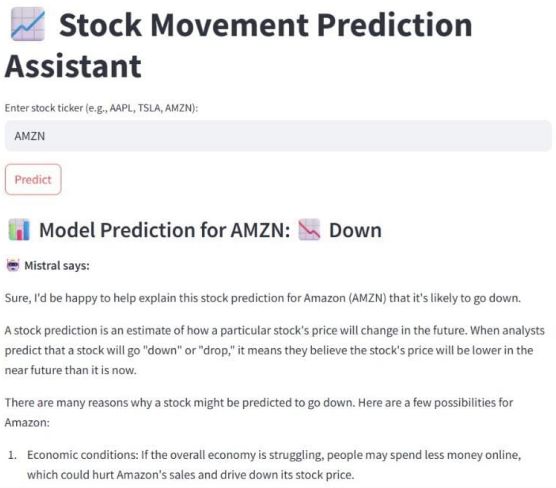

Fig. 5. Streamlit-Based User Interface for Real-Time Stock Prediction and AI Explanation

IV. RESULTS

A. Data Ingestion Performance

In addition to the visual insights, we also evaluated the performance of our data ingestion pipeline as shown in (table-1). For the NewsAPI stream, each batch was typically fetched in 0.4 to 1.3 seconds, with preprocessing times ranging from 0.07 to 0.3 seconds and database insertion times between 0.01 to 0.06 seconds. The full cycle was consistently completed in under 1 second, averaging around 0.8 seconds per batch, with new batches scheduled every 15 minutes. In contrast, Reddit data fetching was extremely fast (approximately 0.0008 to 0.001 seconds), but the preprocessing stage proved more computationally demanding, ranging from 2.7 to 6.5 seconds and occasionally exceeding that. The total Reddit ingestion cycle typically ranged between 3 to 6 seconds, running approximately every minute. While the NewsAPI pipeline proved efficient and stable, Reddit preprocessing emerged as a bottleneck due to its increasing processing time over longer sessions.

TABLE I
DATA INGESTION PERFORMANCE TABLE

Source	Fetch Time	Pre-processing Time	DB Insert Time	Total Time per Cycle	Cycle Frequency
NewsAPI	0.4 - 1.3 s	0.07 - 0.3 s	0.01 - 0.06 s	0.8 s	Every 15 min
RedditAPI	0.0008 - 0.001 s	2.7 - 6.5+ s	0.002 - 0.02 s	3 - 6+ s	Every 1 min

B. PowerBi Dashboard

As part of our results, two Power BI dashboards were developed to visualize both sentiment dynamics and corresponding stock price behavior. The first dashboard, titled Real-Time Financial Sentiment Insights (Figure 1), displays various sentiment analytics extracted from Reddit and news articles. It illustrates the average daily sentiment score for a selected stock ticker, highlights the most frequent sentiment sources (such as news outlets or Reddit users), and classifies sentiment predictions into “Likely Up,” “Likely Down,” and “Neutral.” Additionally, the hourly sentiment heatmap visualizes periods of intense activity, while the stacked area and line charts show prediction counts over time. Further visualizations include average sentiment scores by author and the volume of sentiment entries per ticker. The second dashboard, Stock Price Prediction (Figure 2), links sentiment predictions with actual stock market movements. It presents average closing prices by day, a comparison of daily open and close prices, and the distribution of predicted price movements (up, down, or no change) using a donut chart. It also shows the average sentiment score per ticker and overlays daily sentiment activity with actual market values. This dashboard allows us to visually examine the alignment between sentiment fluctuations and real-world stock behavior.

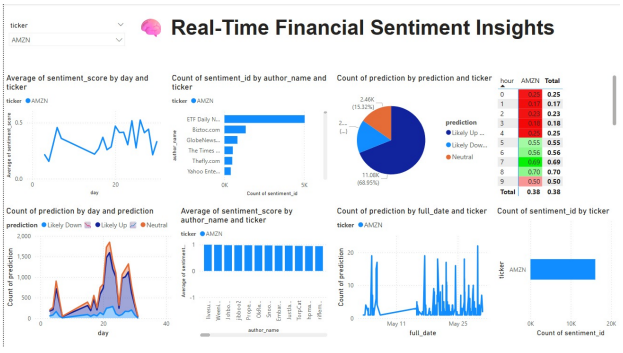

Fig. 6. Real-Time Financial Sentiment Insights

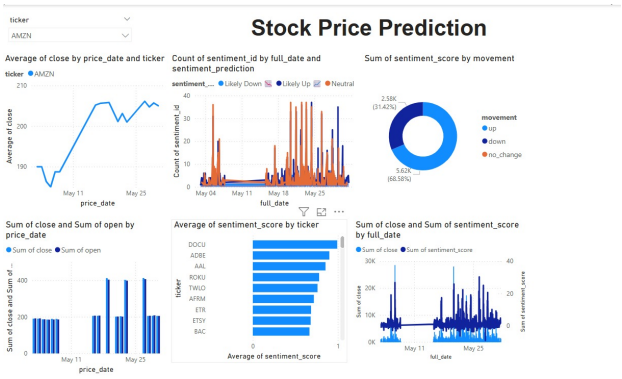

Fig. 7. Real-Time Financial Sentiment Insights

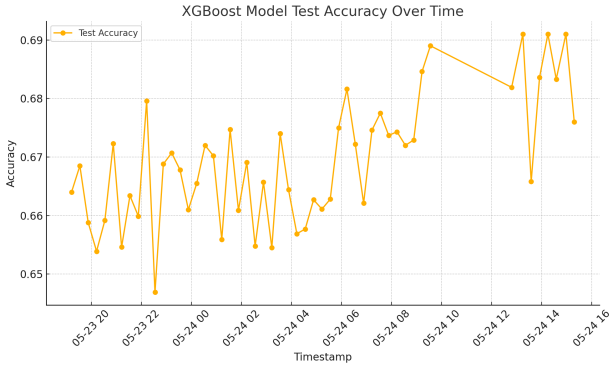

Fig. 8. XGBoost Model Test Accuracy without under sampling

C. Comparative Evaluation of XGBoost With and Without Class Balancing

To evaluate the impact of balancing the dataset, we compared XGBoost model performance before and after applying undersampling as shown in (table 2&3). On the original imbalanced dataset, the model achieved a test accuracy of 0.6551 and showed high recall for the "up" class (0.97) but performed poorly on the "down" class, with a recall of just 0.03. This indicates a strong bias toward the dominant class. After applying undersampling to balance the class distribution, the model's accuracy dropped to 0.5397, but performance across both classes became more balanced, with an F1-score of 0.54 for each class. And it took total of 2.5 minutes in training and testing. As shown in Figures X and Y, undersampling led to a fairer model at the cost of overall accuracy, demonstrating the trade-off between class balance and predictive strength in minority classes.

TABLE II
XGBoost Performance Before Undersampling

Class	Precision	Recall	F1-Score	Support
down	0.36	0.03	0.06	960
up	0.66	0.97	0.79	1896
accuracy	-	-	0.66	2856
macro avg	0.51	0.50	0.42	2856
weighted avg	0.56	0.66	0.54	2856

TABLE III
XGBoost Performance After Undersampling

Class	Precision	Recall	F1-Score	Support
down	0.55	0.55	0.55	980
up	0.53	0.53	0.53	934
accuracy	-	-	0.54	1914
macro avg	0.54	0.54	0.54	1914
weighted avg	0.54	0.54	0.54	1914

D. Performance of Hybrid Ensemble Model After Undersampling

We also evaluated a hybrid ensemble model combining XGBoost, Logistic Regression, and Random Forest. After applying undersampling, the ensemble model achieved a test accuracy of 0.5413, which is comparable to the standalone undersampled XGBoost. Unlike the pre-undersampled model, the ensemble maintained balanced precision and recall values across both classes (down and up), each around 0.54–0.55. This suggests the hybrid approach achieved a stable trade-off between class balance and predictive performance (see Table X).We also evaluated a hybrid ensemble model combining XGBoost, Logistic Regression, and Random Forest. After applying undersampling, the ensemble model achieved a test accuracy of 0.5413, which is comparable to the standalone undersampled XGBoost. Unlike the pre-undersampled model, the ensemble maintained balanced precision and recall values across both classes (down and up), each around 0.54–0.55. This suggests the hybrid approach achieved a stable trade-off between class balance and predictive performance (see Table 5).

TABLE IV
Hybrid Ensemble Model Performance (XGBoost + Logistic Regression + Random Forest)

Class	Precision	Recall	F1-Score	Support
down	0.53	0.54	0.53	934
up	0.55	0.55	0.55	980
accuracy	-	-	0.54	1914
macro avg	0.54	0.54	0.54	1914
weighted avg	0.54	0.54	0.54	1914

E. BERT model performance

The BERT model was trained for 17 epochs to classify stock price movements based on sentiment and ticker input. As shown in the training plots (Figure X), the model achieved rapid convergence, with training loss decreasing steadily and validation accuracy stabilizing around 64–65% after a few epochs. The final evaluation on the test set yielded an overall accuracy of 65%, with the "down" class achieving higher recall (0.69) compared to the "up" class (0.60). The F1-scores for the two classes were 0.66 and 0.63, respectively, And it took total of 1440 minutes in training and testing. reflecting solid generalization and balanced predictive performance. This makes

BERT the best-performing model among those evaluated in this study (see Table X).

TABLE V
BERT Model Performance for Stock Movement Prediction

Class	Precision	Recall	F1-Score	Support
down	0.63	0.69	0.66	1191
up	0.66	0.60	0.63	1191
accuracy	-	-	0.65	2382
macro avg	0.65	0.65	0.64	2382
weighted avg	0.65	0.65	0.64	2382

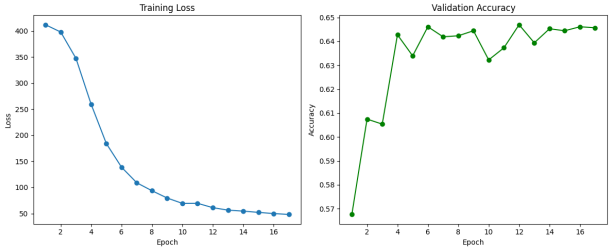

Fig. 9. BERT model performance

V. DISCUSSION

Our proposed system demonstrates significant improvements over prior approaches to stock prediction through real-time sentiment analysis. Compared to early studies such as Bollen et al. [6], which relied solely on Twitter mood to predict market trends, our work incorporates both Reddit and NewsAPI sentiment streams with real-time updates, enabling higher responsiveness. Prior work often utilized static datasets or focused on a single source of sentiment, whereas we addressed this limitation by integrating multiple dynamic sources updated at regular intervals — Reddit every 1 minute and News every 15 minutes.

FinBERT, introduced by Araci [11], and similar transformer-based sentiment models like those explored by Yang and Mo [12] showcased promising performance. However, these models were trained on static, domain-specific corpora. In contrast, our implementation of BERT operates on continually updated data and achieves a 65% classification accuracy, surpassing our hybrid ensemble models (XGBoost + Logistic + Random Forest) which plateaued at 54%. Our BERT model also displayed a more balanced performance between the "up" and "down" classes compared to the skewed metrics in classical models before undersampling.

Furthermore, many studies in the literature lack integrated dashboards for financial decision support. By employing Power BI, we introduced a visual analytics layer, empowering users with interactive and time-aware insights. Additionally, real-time efficiency metrics underscore our system's practical feasibility — Reddit data pipelines operated with preprocessing times between 2.7s to 6.5s, while NewsAPI cycles averaged under 0.8s. These speeds demonstrate our ability to handle continuous data flows without bottlenecks.

In summary, our model not only aligns with but extends prior work in multiple phases: data diversity, real-time streaming, balanced model training, sentiment time-categorization, and practical visualization. This positions our solution as both technically and operationally superior to many previous studies.

VI. CONCLUSION

This study addressed the research gap in real-time, multi-source sentiment-based stock prediction by developing a comprehensive pipeline that integrates live data from Reddit and NewsAPI, combines it with historical stock prices via the yfinance library, and processes it through a robust modeling and visualization framework. Unlike previous works that relied on static datasets or focused narrowly on a single sentiment source, our approach featured temporal sentiment classification, adaptive retraining every 20 minutes, and an interactive Power BI dashboard. We demonstrated that transformer-based models like BERT achieved superior predictive accuracy (65%) compared to classical ensembles (54%), and our system architecture maintained low-latency processing across all stages. The integration of PostgreSQL for data consolidation and Streamlit for the user interface further enhanced the usability and deployment readiness of the platform. Additionally, we employed a FastAPI backend and used Mistral for future-ready model deployment, setting the foundation for extendable real-time analytics. Future Work will aim at improving model performance through techniques such as attention-based temporal modeling, use of sentiment trend deltas, integration of volume and volatility indicators from financial APIs, and dynamic thresholding based on market regimes. Incorporating a hybrid model architecture (e.g., BERT for sentiment + LSTM for sequential stock behavior) may also enhance accuracy. Furthermore, expanding the feature space to include technical indicators, real-time economic news, and alternative data such as earnings call transcripts could significantly improve prediction robustness and contextual intelligence. Enhancing personalization and explainability in the dashboard will also help build greater trust and insight for end-users.

References

[1] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," J. Comput. Sci., vol. 2, no. 1, pp. 1–8, 2011, doi: https://doi.org/10.1016/j.jocs.2010.12.007.

[2] X. Zhang, H. Fuehres, and P. A. Gloor, "Predicting stock market indicators through Twitter," Procedia - Soc. Behav. Sci., vol. 26, pp. 55–62, 2011, doi: https://doi.org/10.1016/j.sbspro.2011.10.562.

[3] T. H. Nguyen, K. Shirai, and J. Velcin, "Sentiment analysis on social media for stock movement prediction," Expert Syst. Appl., vol. 42, no. 24, pp. 9603–9611, Dec. 2015, doi: https://doi.org/10.1016/j.eswa.2015.07.052.

[4] D. Araci, "FinBERT: Financial Sentiment Analysis with Pre-trained Language Models," arXiv, Aug. 2019, doi: https://doi.org/10.48550/arXiv.1908.10063.

[5] T. Jiang and A. Zeng, "Financial sentiment analysis using FinBERT with application in predicting stock movement," arXiv, Jun. 03, 2023. [Online]. Available: https://arxiv.org/abs/2306.02136

[6] N. O. Adelakun and A. B. Adebisi, "Sentiment Analysis of Financial News Using the BERT Model," ITEGAM-JETIA, vol. 10, no. 48, Jan. 2024, doi: https://doi.org/10.5935/jetia.v10i48.1029.

[7] S. Basak, S. Kar, S. Saha, L. Khaidem, and S. R. Dey, "Predicting the direction of stock market prices using tree-based classifiers," North Am. J. Econ. Finance, vol. 47, pp. 552–567, Jan. 2019, doi: https://doi.org/10.1016/j.najef.2018.06.013.

[8] L. Khaidem, S. Saha, and S. R. Dey, "Predicting the direction of stock market prices using random forest," arXiv, 2016. https://arxiv.org/abs/1605.00003.

[9] H. Daori et al., "Predicting Stock Prices Using the Random Forest Classifier," Research Square, Nov. 2022, doi: https://doi.org/10.21203/rs.3.rs-2266733/v1.

[10] M. Vijh et al., "Stock Closing Price Prediction using Machine Learning Techniques," Procedia Comput. Sci., vol. 167, pp. 599–606, 2020, doi: https://doi.org/10.1016/j.procs.2020.03.326.

[11] P. Werawithayaset and S. Tritilanunt, "Stock Closing Price Prediction Using Machine Learning," 2019 ICT&KE, Bangkok, Thailand, 2019, doi: https://doi.org/10.1109/ICTKE47035.2019.8966836.

[12] R. Akita et al., "Deep learning for stock prediction using numerical and textual information," in Proc. IEEE/ACIS ICIS, 2016, doi: 10.1109/ICIS.2016.7550882.

[13] Md. E. Karim and S. Ahmed, "A Deep Learning-Based Approach for Stock Price Prediction Using BiGRU and BiLSTM," ResearchGate, Oct. 2021, doi: https://doi.org/10.1109/GCAT52182.2021.9587895.

[14] J. Maqbool et al., "Stock Prediction by Integrating Sentiment Scores of Financial News and MLP-Regressor: A Machine Learning Approach," Procedia Comput. Sci., vol. 218, pp. 1067–1078, 2023, doi: https://doi.org/10.1016/j.procs.2023.01.086.

[15] D. Duong, T. Nguyen, and M. Dang, "Stock Market Prediction using Financial News Articles on Ho Chi Minh Stock Exchange," IMCOM '16, 2016, doi: https://doi.org/10.1145/2857546.2857619.

[16] M. Sariyer et al., "Individual Stock Price Prediction by Using KAP and Twitter Sentiments with Machine Learning for BIST30," 2022 INISTA, vol. 5, pp. 1–6, Aug. 2022, doi: https://doi.org/10.1109/inista55318.2022.9894172.

[17] X. Li et al., "News impact on stock price return via sentiment analysis," Knowl.-Based Syst., vol. 69, pp. 14–23, 2014, doi: https://doi.org/10.1016/j.knosys.2014.04.022.

[18] E. Teti, M. Dallocchio, and A. Aniasi, "The relationship between twitter and stock prices. Evidence from the US technology industry," Technol. Forecast. Soc. Change, vol. 149, p. 119747, Dec. 2019, doi: https://doi.org/10.1016/j.techfore.2019.119747.

[19] P. Eachempati, P. R. Srivastava, and P. K. Panigrahi, "Sentiment Analysis of COVID-19 Pandemic on the Stock Market," Am. Bus. Rev., vol. 24, no. 1, pp. 141–165, May 2021, doi: https://doi.org/10.37625/abr.24.1.141-165.

[20] V. S. Pagolu et al., "Sentiment analysis of Twitter data for predicting stock market movements," IEEE Xplore, Oct. 01, 2016. https://ieeexplore.ieee.org/abstract/document/7955659.

[21] N. Adlakha, Ridhima, and A. Katal, "Real Time Stock Market Analysis," 2021 Int. Conf. on System, Comput., Autom. and Netw. (ICSCAN), Jul. 2021, doi: https://doi.org/10.1109/icscan53069.2021.9526506.

[22] Y. Yang et al., "Stock Price Prediction Based on XGBoost and LightGBM," E3S Web Conf., vol. 275, p. 01040, 2021, doi: https://doi.org/10.1051/e3sconf/202127501040.

[23] X. Jin and C. Yi, "The Comparison of Stock Price Prediction Based on Linear Regression Model and Machine Learning Scenarios," Atlantis Highlights in Intell. Syst., pp. 837–842, Dec. 2022, doi: https://doi.org/10.2991/978-94-6463-030-5_82.

[24] S. Padmanayana and B. K. Varsha, "Stock Market Prediction Using Twitter Sentiment Analysis," Int. J. Sci. Res. Sci. Technol., pp. 265–270, Jul. 2021, doi: https://doi.org/10.32628/cseit217475.

[25] J. Maqbool et al., "Stock Prediction by Integrating Sentiment Scores of Financial News and MLP-Regressor: A Machine Learning Approach," Procedia Comput. Sci., vol. 218, pp. 1067–1078, 2023, doi: https://doi.org/10.1016/j.procs.2023.01.086.