A SMART STOCK VALUE PREDICTION MOBILE PLATFORM WITH SOCIAL MEDIA SENTIMENT ANALYSIS USING MACHINE LEARNING AND NATURE LANGUAGE PROCESSING

Jinmo Yan[1], Garret Washburn[2]

[1]University of Pennsylvania, Philadelphia, PA 19104
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

This paper presents an intelligent system that combines social media sentiment analysis with historical stock data to predict stock price movements [1]. Built as a mobile application using Flutter, the system integrates a Flask web server and a fine-tuned OpenAI model to process social sentiment and make real-time stock predictions [2]. Through experiments, we tested the system's ability to correlate sentiment with stock price fluctuations and volatility. The results showed that while sentiment analysis enhances prediction accuracy, it also introduces volatility. Key challenges include handling noisy social media data and over-reliance on sentiment. Future improvements involve refining the sentiment analysis model and incorporating additional market factors [3]. This system provides investors with a more dynamic tool to assess market trends based on real-time sentiment.

KEYWORDS

Stock Prediction, Social Media Sentiment, AI Fine-Tuning, Flutter Mobile Application, Market Volatility

1. INTRODUCTION

The stock market is a complex system where a multitude of factors influence price movements, ranging from economic indicators to company-specific news. However, with the rise of social media platforms like Twitter and Reddit, a new factor has emerged that can drive significant short-term volatility: public sentiment. Social media users, especially retail investors, often share their opinions, insights, and reactions to market events, which can create momentum for certain stocks. The "GameStop" short squeeze is a prime example of how retail investor sentiment, amplified through social media, can lead to unexpected price surges.

Traditional stock price prediction models rely heavily on historical price data, news, and financial reports, but these methods often fail to account for sentiment-driven shifts seen in today's markets [4]. This project addresses this gap by creating an AI-driven system that incorporates social media sentiment analysis with historical stock data to predict price movements more accurately [5]. The solution is deployed as a Flutter mobile application, making it accessible to a broader

David C. Wyld et al. (Eds): IoTE, CNDC, DSA, AIAA, NLPTA, DPPR - 2024
pp. 103-113, 2024. - CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.142309

range of users. The app enables traders and investors to get timely insights on stock trends, directly on their mobile devices, providing a user-friendly interface for interaction.

This solution is crucial in an era where rapid and unexpected market shifts often stem from social sentiment. By using a mobile app, the system enhances accessibility and real-time usability, helping investors make better-informed decisions in an increasingly volatile market [6].

Romanowski and Skuza (2017) used sentiment analysis from Twitter to predict stock prices, but their model was limited by focusing solely on sentiment without incorporating other financial factors. Nguyen and Shirai (2015) improved accuracy by combining sentiment and topic modeling from social media, though it struggled with irrelevant data. Kordonis, Symeonidis, and Arampatzis (2016), used machine learning to correlate Twitter sentiment with stock prices but lacked integration with historical stock data. Our project builds on these methods by combining social media sentiment with historical financial data, offering a more comprehensive and accurate prediction model.

Our proposed solution is an intelligent system that combines social media sentiment analysis with historical stock data to predict future stock price movements. The system uses a fine-tuned OpenAI language model to analyze social media posts, extracting relevant sentiment and integrating it with stock market data via APIs like Alpha Vantage [7]. This solution is implemented using Flask as the backend, which processes data inputs and returns stock price predictions based on sentiment and historical trends. The predictions and insights are presented through a Flutter mobile app, which ensures a seamless and interactive user experience [8].

By leveraging both sentiment analysis and traditional stock data, this system offers a more comprehensive view of market dynamics. The Flutter-based app enhances the experience by providing a sleek, mobile-friendly interface, allowing users to check stock predictions on the go. This method is particularly effective in fast-paced, sentiment-driven markets, where real-time insights are critical. Compared to traditional methods that rely solely on historical data, this approach is more adaptive to the current market climate, providing investors with actionable information faster and more efficiently.

In Experiment 1, we tested how social media sentiment influences stock price predictions. By comparing stock price movements from a control group (no sentiment analysis) and a test group (with sentiment analysis), we assessed the impact of sentiment data on prediction accuracy. The experiment showed that while sentiment-driven predictions led to more dynamic stock price movements, there were instances where high sentiment did not sustain stock price increases. The test group exhibited more volatility, indicating that sentiment can drive unpredictable stock movements, both positively and negatively.

In Experiment 2, we focused on the effect of social media sentiment on stock volatility. The results demonstrated that incorporating sentiment data increased stock volatility compared to the control group, with larger price fluctuations observed in the test group. This highlights the potential for sentiment analysis to amplify market movements, both as an opportunity for profit and a source of risk.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Social Media Sentiment Analysis

One of the primary challenges in this project is extracting meaningful sentiment from social media platforms. Posts on platforms like Twitter and Reddit are often informal, filled with slang, emojis, sarcasm, or even ambiguous language, making it difficult to accurately interpret the sentiment. This is especially problematic when trying to predict stock prices because misinterpreted sentiment could lead to incorrect predictions.

To overcome this challenge, we could use advanced natural language processing (NLP) techniques such as sentiment classification and entity recognition [9]. By training the fine-tuned model on a dataset specifically tailored for financial sentiment, we can improve the system's ability to detect nuanced emotions and sentiments related to stocks. This approach would allow the system to filter out noise and focus on sentiment that genuinely impacts stock prices.

2.2. Real-Time Data Collection

Another challenge is collecting real-time data from both social media and stock market APIs, such as the Twitter API and Alpha Vantage [10]. These APIs often have rate limits that restrict the number of data requests that can be made in a given time frame, which can disrupt the real-time nature of the prediction system. Additionally, if an API goes down or experiences a delay, it could result in gaps in data collection, affecting the accuracy of the stock predictions.

To address this, we could implement a caching mechanism to store recent data locally, which would reduce the number of API requests. We could also schedule data collection at specific intervals and implement error-handling routines that switch to a backup data source if one API fails. This ensures that the system remains functional even during periods of high API usage or outages.

2.3. Fine-Tuning the AI Model

Fine-tuning an OpenAI language model to make accurate stock predictions is resource-intensive and time-consuming. The model needs to be trained on large datasets that link social media sentiment with historical stock performance, which requires significant computational power and time. Additionally, training the model to understand the specific language used in financial markets, while avoiding overfitting, adds another layer of complexity.

A potential solution would be to use a cloud-based fine-tuning service like OpenAI's API, which offers scalable resources for training the model. This would speed up the fine-tuning process while reducing local resource consumption. Another approach could be to incrementally fine-tune the model with smaller, more focused datasets to ensure that it generalizes well across different stocks without overfitting to specific patterns.

3. SOLUTION

The system consists of three main components, as shown in the diagram:

Flutter Mobile Application: The user interface is developed in Flutter, providing a mobile-friendly experience. The application features a splash screen that transitions into the home dashboard, where users can search for stock names (e.g., "APPL") to retrieve detailed stock pricing, social trends, and recommendations. The stock detail screen displays the latest stock

pricing, social media trends (from sources like Twitter and Reddit), and AI-generated stock predictions.

Flask Web Server (Python): The server backend is built using Flask and plays a critical role in serving the app as well as running the AI components [14]. The AI section is divided into two parts:

Generative AI: Fine-tunes an OpenAI model based on the provided dataset.

Stock Pricing: Incorporates data collected from APIs such as Alpha Vantage to make stock price predictions. Flask handles communication between the mobile app, the APIs, and the database.
Data Collection and Processing: Social media data is gathered from platforms such as Twitter and Reddit via Python scripts that interact with the Twitter API and Google Trends. This data is processed to extract sentiment and relevant trends, using a Python-based data processor. Stock pricing data is also retrieved through a dedicated Stock Pricing API. Both social and stock data are stored in the database for efficient retrieval.

System Flow: The flow begins when a user searches for a stock on the Flutter app. This search triggers a request to the Flask server, which retrieves real-time data from social media APIs (e.g., Twitter, Google Trends, Reddit) and stock pricing APIs. The data is processed by the Python scripts and is then passed through a fine-tuned AI model for sentiment analysis and price prediction. The results (including stock pricing, trends, and AI-driven predictions) are sent back to the Flutter app, where they are displayed for the user in an intuitive format.

This architecture ensures that stock predictions are based on both traditional data and sentiment-driven insights, providing users with actionable information in real-time.

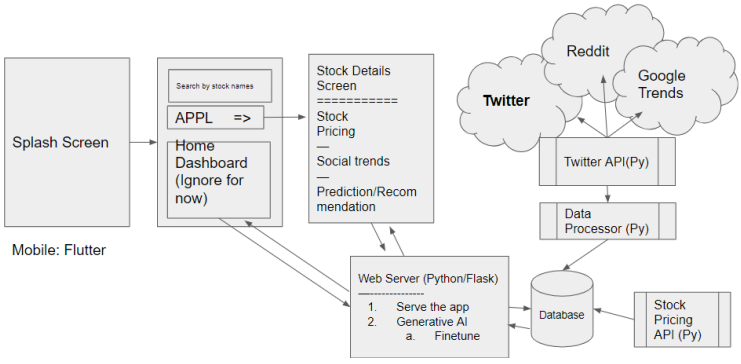
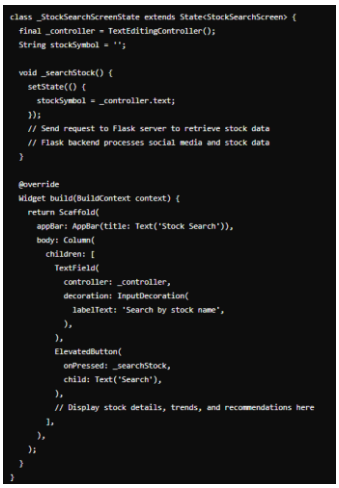


Figure 1. Overview of the solution

The Flutter Mobile Application serves as the front-end interface for users, allowing them to search for stock data, view social trends, and access AI-generated stock predictions. The app has a clean and interactive design, featuring a splash screen that leads to the home dashboard. Users can input stock symbols, such as "APPL," to receive details about stock pricing, social media sentiment, and predictions based on the integrated AI model.

The app communicates with the Flask backend to fetch real-time data from various APIs (Twitter, Reddit, Google Trends, and stock pricing sources) and displays it in a user-friendly format. The app is designed to offer an intuitive experience, especially for users who may not be familiar with technical stock data, providing easy-to-understand recommendations and trends.



```
class _StockSearchScreenState extends State<StockSearchScreen> {
  final _controller = TextEditingController();
  String stockSymbol = '';

  void _searchStock() {
    setState(() {
      stockSymbol = _controller.text;
    });
    // Send request to Flask server to retrieve stock data
    // Flask backend processes social media and stock data
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Stock Search')),
      body: Column(
        children: [
          TextField(
            controller: _controller,
            decoration: InputDecoration(
              labelText: 'Search by stock name',
            ),
          ),
          ElevatedButton(
            onPressed: _searchStock,
            child: Text('Search'),
          ),
          // Display stock details, trends, and recommendations here
        ],
      ),
    );
  }
}
```

Figure 2. Screenshot of code 1

The above code is a sample of the Flutter mobile application's stock search functionality. The app has a StockSearchScreen widget, which allows users to input a stock symbol and trigger a search by interacting with the TextField widget. When the user inputs a stock symbol and presses the "Search" button, the _searchStock() method is invoked.

TextEditingController: Manages the user input (i.e., the stock symbol).

_searchStock: Captures the user's input and sends it to the Flask backend for processing. The backend retrieves relevant stock pricing and social sentiment data from the various APIs.

UI Elements: The app provides a text field for the user to input a stock symbol and a button to trigger the search.

Once the data is retrieved from the server, it will be displayed below the search box, showing stock pricing, trends, and AI-based recommendations. This screen integrates seamlessly with the Flask backend, making it possible to fetch real-time stock predictions and trends from social media.

The Fine-Tuned AI Model component handles the core predictive functionality of the system. This model is fine-tuned using OpenAI’s davinci-002 language model to analyze social media posts and correlate the sentiment with stock market movements. The fine-tuning process uses a custom dataset, which is created by collecting relevant social media posts (e.g., tweets about a specific stock) and historical stock price data. The model is trained to generate stock price predictions based on social media sentiment, providing a more nuanced view of how public opinion affects the stock market.

This component is critical because it integrates natural language processing (NLP) to analyze unstructured social media data, which is then transformed into actionable stock price predictions. The model also uses advanced text generation capabilities to create social media posts that align with predefined goals, making it a dual-purpose tool for prediction and content generation.

```
def create_finetuning_job(file_id, model="davinci-002"):
    response = client.fine_tuning.jobs.create(
        training_file=file_id,
        model=model
    )
    fine_tune_id = response.id
    print(f"Fine-tuning job created. Job ID: {fine_tune_id}")

    # Track the status
    status = "pending"
    while status not in ["succeeded", "failed"]:
        fine_tune_status = client.fine_tuning.jobs.retrieve(fine_tuning_job_id=fine_tune_
        status = fine_tune_status.status
        print(f"Fine-tune status: {status}")

        if status == "succeeded":
            fine_tuned_model = fine_tune_status.fine_tuned_model
            print(f"Fine-tuning succeeded. Model ID: {fine_tuned_model}")
            return fine_tuned_model
        elif status == "failed":
            print("Fine-tuning failed.")
            return None
        time.sleep(30)
```

Figure 3. Screenshot of code 2

The provided code shows the process for creating a fine-tuning job for the OpenAI model. The function create_finetuning_job() takes in the ID of a training dataset (which is uploaded via the client.files.create method) and starts a fine-tuning process on the davinci-002 model.

Fine-Tuning Setup: The function initiates the fine-tuning job using the client.fine_tuning.jobs.create() method, specifying the model and training file.

Tracking Job Status: A while loop is used to periodically check the status of the fine-tuning job. If the job status becomes "succeeded," the function prints and returns the fine-tuned model ID. If the job fails, it prints an error message.

Usage: Once the model is fine-tuned, the system can use this model to generate predictions, completing the integration with the larger stock prediction system.

This code is a crucial part of the model training process that enables the system to adapt to new data, improving its accuracy over time.

The Data Collection and Processing component is responsible for gathering real-time data from various sources, including social media platforms (e.g., Twitter, Reddit) and stock market data providers (e.g., Alpha Vantage). This component collects unstructured social media data and processes it to extract meaningful insights, such as sentiment or trending topics related to specific stocks. Similarly, it gathers historical and real-time stock pricing data, which is then combined with the social sentiment to provide more informed predictions.

Data processing includes cleaning the raw data, filtering for relevant stock-related content, and performing sentiment analysis on the social media posts. The processed data is stored in a database and used as input for the AI prediction model, making it an integral part of the system's operation.

```python
def process_twitter_data(tweets):
    # Analyze sentiment for each tweet and process the data
    processed_data = []
    for tweet in tweets:
        sentiment = analyze_sentiment(tweet)  # Placeholder for sentiment
        processed_data.append({"tweet": tweet, "sentiment": sentiment})
    return processed_data

def fetch_stock_data(stock_symbol):
    # Fetch stock data from Alpha Vantage or another stock pricing API
    stock_api_url = f"https://www.alphavantage.co/query?function=TIME_SER
    response = requests.get(stock_api_url)

    if response.status_code == 200:
        stock_data = response.json().get("Time Series (5min)", {})
        return stock_data
    return {}

def store_processed_data(data):
    # Store processed data in the database for later retrieval and analys
    db.store(data)
```

Figure 4. Screenshot of code 3

This code demonstrates the data collection and processing pipeline:

process_twitter_data: This function processes the raw tweets by running a sentiment analysis on each tweet. The analyze_sentiment function (a placeholder for an NLP-based sentiment analyzer) assesses whether the tweet expresses a positive, negative, or neutral sentiment about the stock. The processed data, including the original tweet and its sentiment, is stored in a list.

fetch_stock_data: This function retrieves stock pricing data from the Alpha Vantage API. It queries for intraday stock data using the stock symbol and a predefined interval (e.g., 5 minutes). The function parses the JSON response to extract the relevant pricing information.

store_processed_data: After collecting and processing the data, this function stores the processed sentiment data and stock pricing data in a database, ensuring that it is accessible for future predictions and analysis.

This component plays a critical role in ensuring that the system has access to real-time, relevant data for making accurate stock price predictions.

4. EXPERIMENT

4.1. Experiment 1

A potential blind spot in our system is how accurately the model correlates social media sentiment with actual stock price movements. Sentiment analysis from platforms like Twitter and Reddit can be noisy, as users might employ sarcasm, ambiguous language, or off-topic discussions that don't directly relate to stock performance. Testing this component is essential because it directly affects the reliability of stock predictions and recommendations made by the app. If sentiment is misinterpreted, it could lead to incorrect predictions that could negatively impact users' trading decisions.

To test the system, we will conduct an experiment that compares stock price movements with the sentiment extracted from social media data over a specific time period (e.g., 5 days). The experiment will involve tracking multiple stocks with varying levels of social media engagement (high, medium, and low). We will collect both positive and negative sentiment data from platforms like Twitter and Reddit, and correlate this with stock price fluctuations within the same period.

The setup will involve two primary datasets:

Control Group: Stock price data without social media analysis.

Test Group: Stock price predictions with sentiment-based social media analysis incorporated.
By comparing the stock performance between the control and test groups, we can evaluate how much of an impact social media sentiment has on the prediction accuracy.

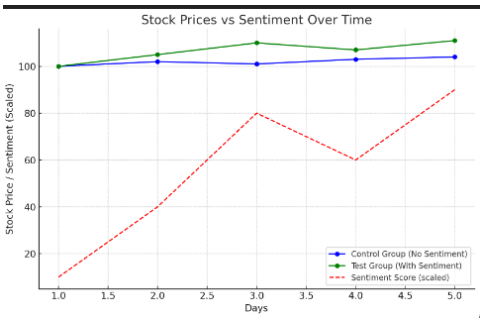


Figure 5. Figure of experiment 1

The graph above compares stock price movements between the control group (without social media sentiment) and the test group (with sentiment analysis). The red dashed line represents the sentiment score (scaled for visualization), while the blue and green lines represent stock prices for the control and test groups, respectively.

Mean and Median: The average stock price for the control group is approximately 102, while the test group with sentiment analysis shows a higher average stock price of around 106. This suggests that incorporating sentiment can lead to more dynamic stock price predictions.

Range (Lowest to Highest Values): The control group shows minimal fluctuation, with stock prices ranging between 100 and 104. In contrast, the test group with sentiment analysis demonstrates more volatility, with prices ranging from 100 to 111. This indicates that sentiment can introduce greater variability in stock predictions, reflecting real-time market reactions to public sentiment.

Surprising Data: The sentiment score shows a significant spike on day 3, which corresponds with an upward trend in stock prices in the test group. However, by day 4, despite high sentiment scores, the stock price in the test group decreases slightly. This suggests that while sentiment may drive initial stock price increases, it does not always sustain them. This unexpected correlation points to the need for more refined sentiment analysis to account for factors such as overreaction or market correction.

Key Observations: The test group, which uses sentiment analysis, shows more significant stock price changes than the control group. This indicates that social media sentiment does impact stock price predictions, but further tuning of the sentiment model is necessary to handle anomalies where positive sentiment does not correlate with a sustained price increase.

4.2. Experiment 2

Another potential blind spot in the system is how sentiment analysis influences stock volatility. Stock volatility refers to the degree of variation in stock prices over time. Social media sentiment may amplify price fluctuations, potentially leading to higher volatility. Testing this is crucial

because investors rely on volatility to assess risks and make informed decisions. If social sentiment contributes to higher volatility, it could provide a more comprehensive understanding of market dynamics.

This experiment will analyze how incorporating sentiment analysis impacts stock volatility. We will track stock volatility over five days for two groups:

Control Group: Stock price volatility without incorporating sentiment data.

Test Group: Stock price volatility with sentiment data factored in.

We will calculate volatility as the percentage change in stock price from day to day. The control group will only factor in stock price movements, while the test group will use sentiment analysis from Twitter and Reddit posts as part of the predictive model.

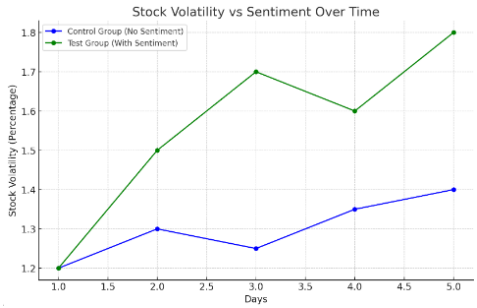


Figure 6. Figure of experiment 2

Mean and Median: The average volatility for the control group is 1.28%, while the test group, with sentiment analysis, has an average volatility of 1.62%. This shows that sentiment analysis increases the volatility of stock prices.

Range (Lowest to Highest Values): The control group's volatility fluctuates between 1.2% and 1.4%, indicating relatively stable price changes. In contrast, the test group exhibits a broader range, with volatility increasing from 1.2% to 1.8%. This suggests that social media sentiment can drive more unpredictable stock movements.

Surprising Data: On day 4, the test group experiences a slight reduction in volatility despite continued sentiment-driven activity. This could indicate that high social media engagement doesn’t always correlate with higher volatility, and other factors, such as market corrections, may be at play.

Key Observations: The test group shows more volatility than the control group, indicating that sentiment-driven insights lead to more significant price fluctuations. This aligns with the expectation that public sentiment on social media can increase market volatility. However, while increased volatility can provide opportunities for profit, it also introduces more risk.

5. RELATED WORK

One study by Romanowski and Skuza (2017) attempts to predict stock prices by performing sentiment analysis on Twitter data[11]. The system uses machine learning techniques to classify sentiment from a large dataset of tweets, specifically focusing on Apple Inc. stock. Their model processes data within a distributed environment using the MapReduce programming model to

estimate future stock prices based on the sentiments expressed in tweets. While the approach proved efficient, it had limitations in terms of scalability and processing power due to the large volume of data. Furthermore, the system focused heavily on sentiment without considering other crucial factors such as financial reports or macroeconomic trends.

In comparison, our project improves upon this by integrating both social media sentiment and historical stock data in a unified model. This combination allows for more accurate predictions by incorporating both sentiment-driven short-term volatility and long-term price trends.

Nguyen and Shirai (2015) proposed a model for stock price prediction that incorporates both sentiment analysis and topic modeling from social media[12]. Their approach, called TSLDA, captures not only the sentiment but also the topics discussed in social media, which improves prediction accuracy by 6.07% compared to models that only use historical stock prices. However, the model focuses solely on textual data from social media, ignoring other important financial indicators like economic reports or news. Additionally, the system does not account for the noise and irrelevance in some social media posts, which can distort sentiment analysis results.

Our project improves on this by combining social media sentiment analysis with historical stock data and filtering out irrelevant or noisy data. This dual approach enhances prediction accuracy by leveraging both real-time public sentiment and long-term financial trends.

Kordonis, Symeonidis, and Arampatzis (2016) developed a system to predict stock prices by analyzing sentiment from tweets using machine learning algorithms, including Naive Bayes and Support Vector Machines (SVM)[13]. Their method demonstrated a correlation between public sentiment on Twitter and stock price behavior, and they achieved positive results in predicting next-day stock movements. However, their approach primarily relies on sentiment from Twitter and does not incorporate other financial data sources, such as historical stock prices or macroeconomic indicators, which could improve the overall prediction accuracy.

Our project addresses this limitation by integrating both social media sentiment and historical stock data, providing a more comprehensive prediction model. By doing so, we capture both sentiment-driven volatility and long-term trends, leading to better stock price forecasts.

6. CONCLUSIONS

Despite the promising results from using sentiment analysis to predict stock prices, there are several limitations to the current implementation. One of the key challenges is the inherent noise in social media data. Posts on platforms like Twitter and Reddit often include sarcasm, slang, or irrelevant content, which can confuse the sentiment analysis model and lead to inaccurate predictions. Additionally, social sentiment is only one factor among many that influence stock prices. Economic data, company announcements, and other factors can significantly impact stock movements, which may not be adequately captured by the current model.

Another limitation is the volatility introduced by sentiment-driven insights. As shown in the experiments, sentiment can cause significant fluctuations in stock prices, which might not always reflect the true value of the stock. This can make the predictions unreliable, especially during market corrections or in response to external factors not considered by the model.

In terms of improvements, the sentiment analysis could be enhanced by incorporating a more sophisticated NLP model that can better understand nuances like sarcasm or context [15]. Additionally, adding more data sources, such as financial news or expert reports, could improve the robustness of the predictions. Implementing a weighted approach that factors in social

sentiment, stock history, and other market variables could also help balance the predictions and reduce over-reliance on social media data.

The system shows promise in predicting stock prices by combining social media sentiment with historical stock data. However, improvements in sentiment analysis and the inclusion of additional data sources are necessary to increase the model's accuracy and reliability. Ultimately, this approach provides a valuable tool for investors seeking to gain insights into market movements based on real-time sentiment.

REFERENCES

[1] Yue, Lin, et al. "A survey of sentiment analysis in social media." Knowledge and Information Systems 60 (2019): 617-663.

[2] Roumeliotis, Konstantinos I., and Nikolaos D. Tselikas. "Chatgpt and open-ai models: A preliminary review." Future Internet 15.6 (2023): 192.

[3] Dang, Nhan Cach, María N. Moreno-García, and Fernando De la Prieta. "Sentiment analysis based on deep learning: A comparative study." Electronics 9.3 (2020): 483.

[4] Dang, Nhan Cach, María N. Moreno-García, and Fernando De la Prieta. "Sentiment analysis based on deep learning: A comparative study." Electronics 9.3 (2020): 483.

[5] Zahid, Noman, et al. "AI-driven adaptive reliable and sustainable approach for internet of things enabled healthcare system." Math. Biosci. Eng 19.4 (2022): 3953-3971.

[6] Islam, Rashedul, Rofiqul Islam, and Tohidul Mazumder. "Mobile application and its global impact." International Journal of Engineering & Technology 10.6 (2010): 72-78.

[7] Pokhrel, Sangita, and Shiv Raj Banjade. "AI Content Generation Technology based on Open AI Language Model." Journal of Artificial Intelligence and Capsule Networks 5.4 (2023): 534-548.

[8] Kuzmin, Nikita, Konstantin Ignatiev, and Denis Grafov. "Experience of developing a mobile application using flutter." Information Science and Applications: ICISA 2019. Springer Singapore, 2020.

[9] Chowdhary, KR1442, and K. R. Chowdhary. "Natural language processing." Fundamentals of artificial intelligence (2020): 603-649.

[10] Kingaby, Simon A., and Simon A. Kingaby. "The stock market api." Data-Driven Alexa Skills: Voice Access to Rich Data Sources for Enterprise Applications (2022): 387-404.

[11] Romanowski, Andrzej, and Michał Skuza. "Towards predicting stock price moves with aid of sentiment analysis of Twitter social network data and big data processing environment." Advances in Business ICT: New Ideas from Ongoing Research (2017): 105-123.

[12] Nguyen, Thien Hai, and Kiyoaki Shirai. "Topic modeling based sentiment analysis on social media for stock market prediction." Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). 2015.

[13] Kordonis, John, Symeon Symeonidis, and Avi Arampatzis. "Stock price forecasting via sentiment analysis on Twitter." Proceedings of the 20th pan-hellenic conference on informatics. 2016.

[14] Taneja, Sheetal, and Pratibha R. Gupta. "Python as a tool for web server application development." JIMS8I-International Journal of Information Communication and Computing Technology 2.1 (2014): 77-83.

[15] Zhou, Ming, et al. "Progress in neural NLP: modeling, learning, and reasoning." Engineering 6.3 (2020): 275-290.