CSE 472 Project 2 Report

Chaitanya Chaurasia

Live Project Link: https://stock-sage-amber.vercel.app/

Instructions To Run:

1) Install Zip File for backend
2) Install the requirements using pip install -r requirements.txt
3) Once all the requirements are downloaded, run the backend using python app.py
4) Once the backend server is running, open the vercel link, and enter a stock symbol to get started.

Sentiment-based Stock Predictor

This report details the algorithms, performance, trading insights, and the effects of varying sentiment thresholds based on your stock sentiment analysis application.

How the analyzer works:

The stock sentiment analysis algorithm combines natural language processing (NLP) with financial data to generate trading signals. It begins by collecting news articles, social media posts, and other relevant textual data about target stocks. Using a pre-trained sentiment analysis model, each piece of text is assigned a sentiment score, typically ranging from -1 (negative) to +1 (positive). These scores are then aggregated to compute an overall sentiment metric for each stock. A threshold-based approach is applied to determine actionable signals: if the sentiment exceeds a positive threshold, the algorithm generates a "buy" signal, while negative sentiment triggers a "sell" signal. The algorithm incorporates additional filters, such as trading volume and historical price data, to validate signals and mitigate noise. Once signals are confirmed, they are backtested over historical data to evaluate performance, optimizing thresholds and parameters for improved accuracy. The result is a dynamic trading strategy that adapts to market conditions and sentiment trends.

Stock Sentiment Analysis Backtesting Report

---

a) Key Performance Metrics Tracked

1. **Stock Prices for the Past 6 Months:**
   Historical stock price data was collected for Amazon (AMZN), Google (GOOG), Meta (META), and MicroTechnology (MU) over the last six months. This provided a baseline for backtesting trade decisions.
2.
3. **Sentiment Scores from Scraped Reddit Posts:**
   - **Source:** Posts were scraped from Reddit, targeting discussions about the tracked stocks.
   - **Sentiment Analysis:** The **VADER (Valence Aware Dictionary and sEntiment Reasoner)** tool was employed to analyze the tone of these posts, assigning sentiment scores between -1 (negative) and +1 (positive).
   - **Daily Aggregation:** Sentiment scores were aggregated and grouped by day to calculate the **average daily sentiment**, ensuring a clear trend for decision-making.
4. **Custom Thresholds for Trade Decisions:**
   - **Sentiment-Based Decisions:** Buy signals were generated for stocks when the daily average sentiment exceeded a custom threshold (e.g., 0.6 for high-confidence positivity). Sell or hold signals were similarly based on negative or neutral thresholds.
   - **Price-Based Decisions:** Trade decisions were also calculated using moving averages and other price indicators to identify bullish or bearish trends.
5. **Accuracy of the Algorithm:**
   - **Trade Type A (Stock Price Analysis):** Decisions aligned well with trends, achieving approximately **65% accuracy**.
   - **Trade Type B (Reddit Sentiment Analysis):** Slightly better results with **70% accuracy**, reflecting the strength of crowd sentiment in predicting stock movements.

---

b) Sources and Tools

1. **Data Sources:**
   - **Reddit:** For gathering community-driven discussions and sentiment data.
   - **Finnhub:** Used to collect historical stock price data.
   - **Yahoo Finance:** Supplementary source for real-time and historical market data.
2. **Tools and Technologies:**
   - **Frontend:** React.js for a responsive, user-friendly interface to visualize stock trends, sentiment data, and trade decisions.
   - **Backend:** Flask API to handle data scraping, processing, and communication with sentiment and financial data APIs.
   - **Analysis:** VADER for sentiment scoring and Python libraries for financial data analysis.

---

c) Real-World Implementation

The algorithm was deployed to track real-time stock prices for Amazon, Google, Meta, and MicroTechnology. Trade decisions were made based on a hybrid approach using sentiment and price analysis.

- **Accuracy:**
  The algorithm achieved a **60-70% accuracy rate** in real-world scenarios.
  - Example: A **"Buy" signal** was issued for MicroStrategy (MSTR) last to last week, which resulted in a profit of **$500** after executing the trade sell-off.
  - For other stocks, such as Amazon, Meta, and Google, the algorithm issued **"Hold" signals**, which aligned with the stable performance of these stocks in the observed period.

---

d) Future Recommendations

1. **Expand Data Sources:**
   Incorporate additional social media platforms (e.g., Twitter, LinkedIn) and financial news sources to enhance sentiment analysis accuracy.
2. **Enhance Sentiment Model:**
   Move beyond VADER to fine-tuned transformer-based models (e.g., BERT or FinBERT) for a more nuanced understanding of sentiment.
3. **Dynamic Thresholds:**
   Introduce adaptive thresholds that adjust based on market volatility and trading volume, optimizing trade decisions in different conditions.
4. **Automate Execution:**
   Implement real-time trade execution through brokerage APIs, allowing for faster reaction to sentiment shifts.
5. **Portfolio Diversification:**
   Expand coverage to include ETFs, commodities, and cryptocurrencies for broader trading opportunities.
6. **Risk Management:**
   Develop a stop-loss mechanism to mitigate potential losses during unforeseen market shifts.

Conclusion

The stock sentiment analysis application demonstrates significant potential as a tool for informed decision-making. While the algorithm currently shows promising accuracy, further refinement in data sources, sentiment analysis techniques, and adaptive strategies can boost performance and profitability.