Sentimental Stock Prediction Analysis

1. Introduction

We underestimated the complexity of real-time sequential data in stock prediction, assuming it was a matter of trend analysis. However, we recognized that stock movements are influenced by both numerical patterns and market sentiment. This led us to design an investor-mimicking system that integrates structured price data (open, close, volume) with unstructured emotional signals from news.

Traditional models like moving averages and regressions lacked adaptability to real-time events, treating all historical data uniformly. To overcome this, we employed LSTM networks capable of learning temporal dependencies and incorporated FinBERT to transform daily news headlines into sentiment vectors. This architecture enabled dynamic, context-aware predictions by fusing quantitative trends with qualitative market sentiment.

2. Gathering Data – What We Chose and Why

With the problem defined, we selected an integrated toolset capable of processing both structured market data and unstructured financial news sentiment. This environment supported end-to-end functionality—data acquisition, transformation, feature engineering, modeling, and visualization—forming the foundational layer of our predictive pipeline.

yFinance and Finnhub:

- We used yfinance to collect six months of stock market data and feature engineered it.
- We captured sentiment by querying the top 5 daily news per stock using the Finnhub API and analyzed them with FinBERT. Each headline was labeled as Positive, Negative, or Neutral, and assigned a corresponding numerical sentiment
  (Positive = +1.0 × Confidence) & (Negative = −1.0 × Confidence) & (Neutral = +0.2 × Confidence)
  Here, Neutral is not zero-impact as it's mildly reassuring, hence a small positive score.
- These per-headline scores were averaged to compute a **Daily Sentiment Score**, which was included as an input feature in our prediction model, acting as an **emotional signal** from financial media.

3. Exploratory Data Analysis

Correlation Matrix: We used a correlation heatmap to study feature relationships. Price-based features (Open, High, Low, Close) showed perfect correlation and helped us identify redundant features. Market Cap and Total Market Cap were also highly correlated (~0.90), validating our broader market metric

Moving Averages: We used 7-day and 30-day moving averages to visualize short and long-term price trends and detect trend.

Autocorrelation (ACF/PACF): These plots revealed that stock prices were influenced by the past 30 days. This directly informed us of our decision to use a **30-day input window** for the LSTM model.

Log Return Distribution: Stock price returns are not normally distributed; they have fat tails and sharp peaks. By analyzing log returns, we identified periods of high volatility, which helped us validate the need for deep learning models instead of traditional regressions.

4. Process Flow

Data Collection > Sentiment Analysis >Feature Engineering >Model Training & Prediction > Streamlit Application

5. Building the Brain – Training and Prediction

We used an LSTM model to capture time-based patterns in 30-day sequences of stock data, where each day included 8 features: **open, high, low, close prices, volume, market cap, total market cap, and a sentiment score.** Initially, a vanilla LSTM learned general trends and predicted the next day's closing price. To improve interpretability and performance, we added two types of attention mechanisms:

- Temporal Attention focused on identifying which days in the 30-day sequence were most important, allowing the

model to emphasize key time points (e.g., days with big price swings or sentiment shifts). **- Feature Attention** was applied across the 8 daily input features, helping the model dynamically assign importance to features like sentiment or volume depending on market context.

Together, these attentions enabled the model to simulate how human investors shift focus some days and some indicators matter more than others. We trained the model using **mean squared error (MSE)** loss and optimized it using the **AdamW optimizer.**

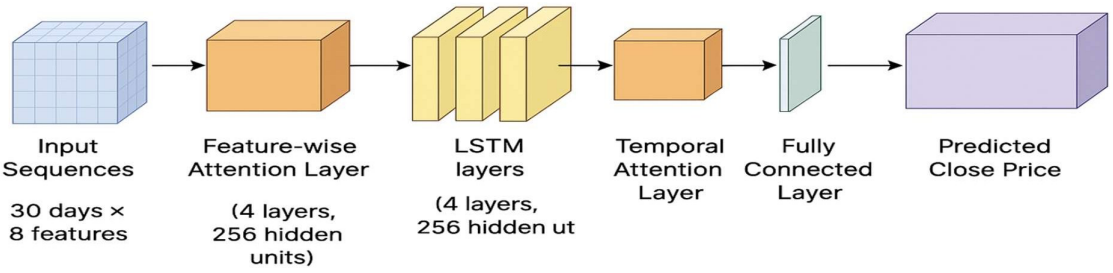

Fig 1: Model Architecture

6. Evaluation Metrics – Understanding Model Accuracy

Once training was complete, we tested the model's ability to predict the next day's closing price using the most recent 30-day sequences. We evaluated the model across different stocks using standard metrics:

- Mean Squared Error (MSE): Penalizes larger errors more heavily.
- Root Mean Squared Error (RMSE): Gives error in the same units as price.
- Mean Absolute Error (MAE): Captures average absolute difference.
- R^2 Score: Measures how well the model explains price variance (closer to 1 is better).

We evaluated several models—including a vanilla LSTM, multi-layered LSTM, and a single attention model—before finalizing a 4-layer LSTM with 256 hidden units and dual attention (feature-level and temporal) using a 30-day sliding window. This architecture mimics how traders weigh signals and recent trends. The attention-augmented model consistently outperformed baselines with lower RMSE/MAE and higher R^2 scores. Visualization shows a strong alignment between actual and predicted prices, confirming the model's forecasting accuracy.

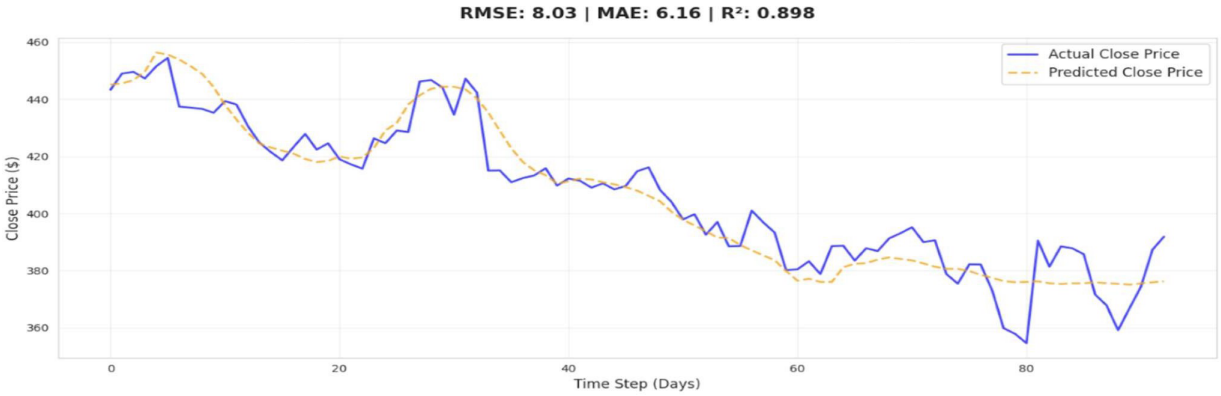

Fig 2: Actual vs Predicted Price of stock

7. AI Investment Strategist – Interactive Dashboard with AI Agents

To bring our model to life, we built an interactive Streamlit dashboard that serves as a real-time investment assistant, integrating deep learning, sentiment analysis, and conversational AI into a seamless user experience.

Behind-the-Scenes Intelligence: The app collects real-time financial data via Yahoo Finance, analyzes recent news using FinBERT for sentiment scoring, and feeds the results into our trained LSTM-Dual Attention-based model. Predictions are based on the most recent 30-day history, incorporating both market indicators and emotional sentiment.

We also integrated Gemini 2.0 Flash via the Agno library, a lightweight agent framework that allows us to create intelligent assistants with specific financial roles:

- The Market Analyst Agent compares stock performances across the last 6 months.
- The Company Research Agent retrieves fundamentals, sector data, and recent news.
- The Stock Strategist Agent evaluates risk-reward profiles and delivers recommendations
- The Team Lead Agent synthesizes everything into a cohesive, ranked investment report.

Dashboard Features

- **Ticker Input & Live Execution:** Users enter two or more stock symbols and the Gemini API key, and the app fetches and processes data in real-time.
- **Generated Report:** The dashboard outputs a structured investment report that includes:
  - Historical stock performance (visualized over 6 months)
  - Predicted next day closing prices
  - News-based sentiment trends
  - Company insights and fundamentals
  - Final stock rankings with AI-generated explanations (e.g., "AAPL marked as a BUY for brand strength and dividend stability.")
- **PDF Export:** A downloadable PDF summarizes the analysis, making it easy to share or revisit investment insights.
- **Reset App – Explore Again:** After generating one report, users can hit the Reset App button to reset the screen.

Fig 3: Dashboard Input Panel: Users enter stock symbols & API key to generate investment reports

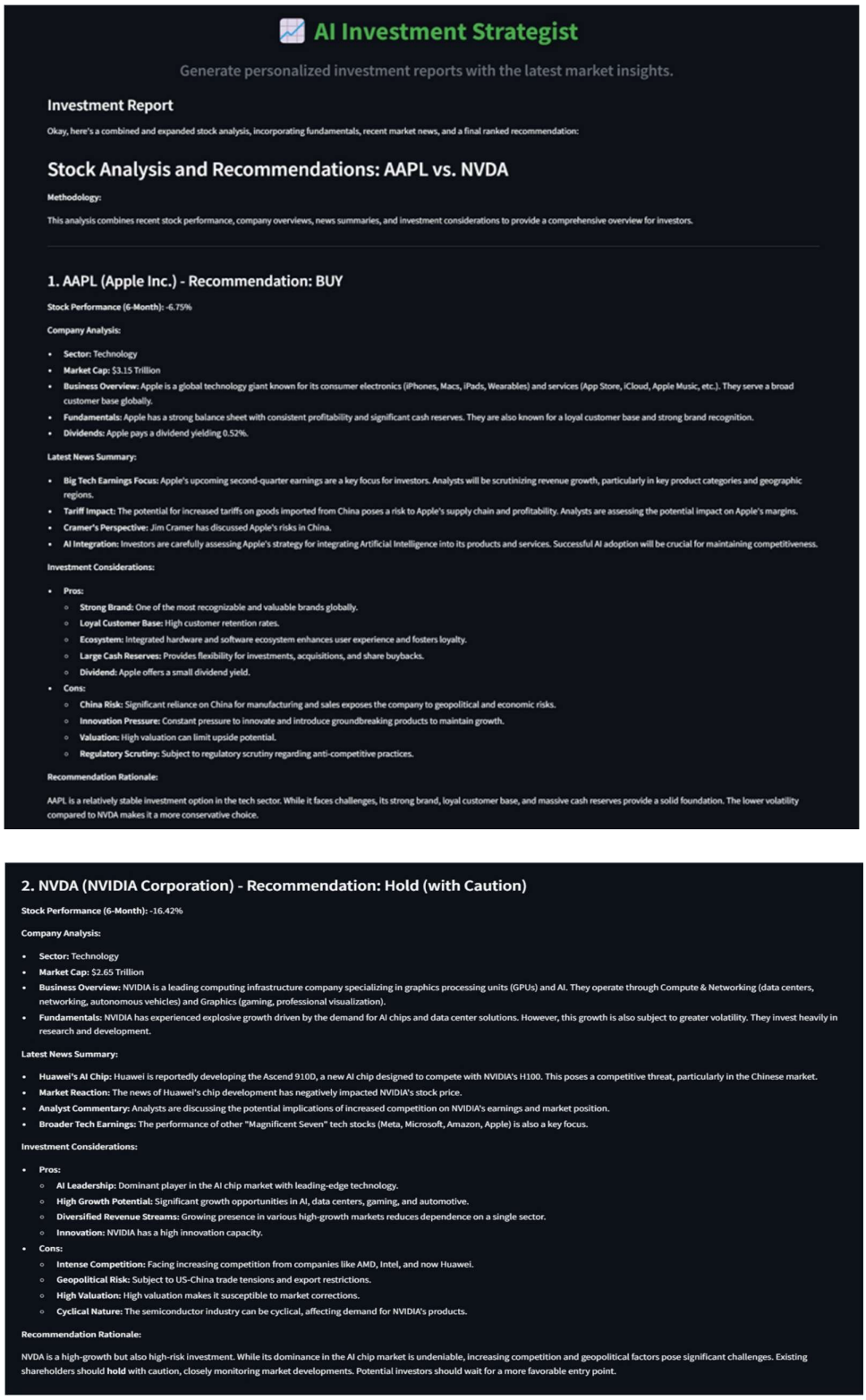

Fig 4: Final Stock Ranking and Recommendation Summary: AI-generated ranking of selected stocks

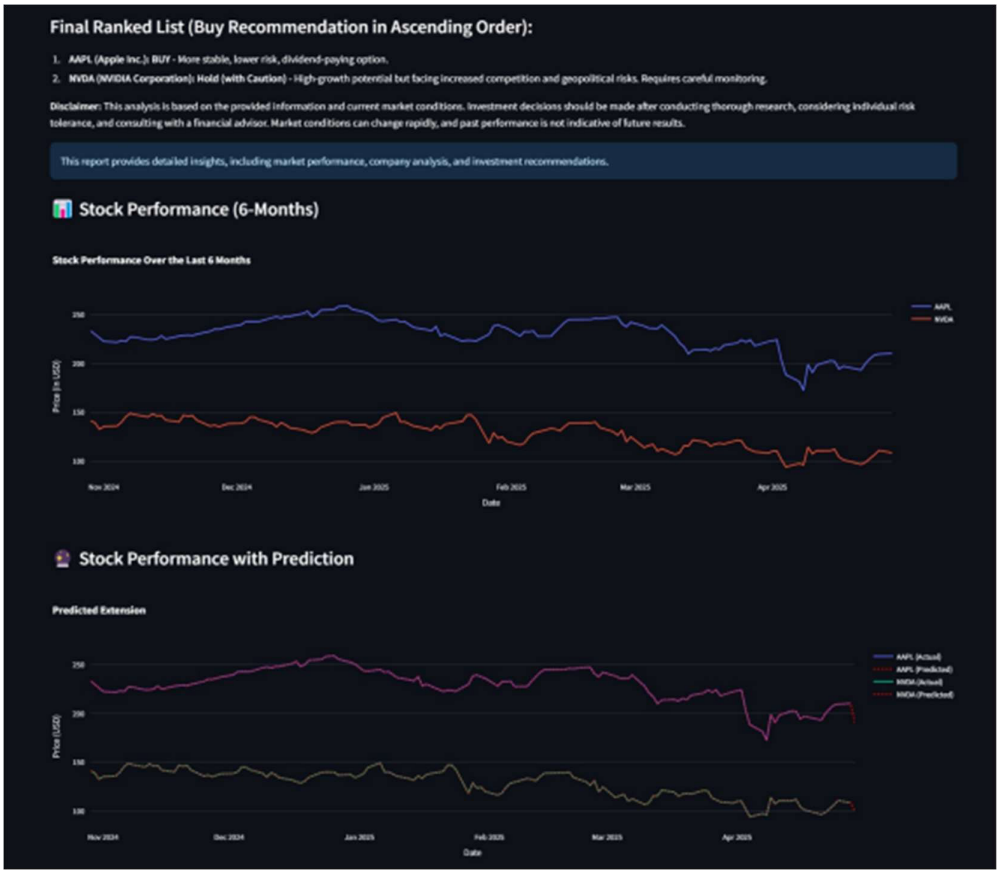

Fig 5: Stock Price Trends and Model Predictions: Actual stock performance over 6 months (top), followed by LSTM-predicted next-day price extensions