Hybrid Deep Learning for S&P 500 Prediction: A Multi-Modal Stacked Ensemble Approach

Pragyan Jyoti Borthakur
Arizona State University
Tempe, Arizona, USA
pborthak@asu.edu

Kavinkarthik Ashok Kumar
Arizona State University
Tempe, Arizona, USA
kavinkarthik.a@asu.edu

Xiao Chang
Arizona State University
Tempe, Arizona, USA
xchang22@asu.edu

Chandana Pulikanti
Arizona State University
Tempe, Arizona, USA
cpulikan@asu.edu

Abstract

Predicting the daily directional movement of the S&P 500 index is a fundamental challenge in quantitative finance, characterized by low signal-to-noise ratios and non-stationary market regimes. This project presents a robust, multi-modal machine learning system designed to forecast market direction by fusing quantitative market data with macroeconomic indicators (VIX, Treasury Yields) and AI-driven sentiment analysis.

The proposed architecture employs a Stacked Generalization strategy, integrating a lightweight decoder-only Transformer (127k parameters) and a Bidirectional LSTM (240k parameters). To mitigate risk, the system incorporates a tri-class labeling approach (Up/Flat/Down) and a conformal-style confidence-thresholding layer designed to selectively abstain from uncertain trades. Experimental results on historical data (2015–2025) demonstrate that the proposed two-model ensemble achieves a directional accuracy of 57.93%, significantly outperforming Random Forest (54.71%) and pure LSTM baselines. When restricting to high-confidence days via conformal prediction, realized accuracy on traded days increases further into the mid-60% range at the cost of lower coverage.

CCS Concepts

• **Computing methodologies** → **Neural networks**; • **Applied computing** → *Economics.*

Keywords

Stock Prediction, Transformers, LSTM, Ensemble Learning, Sentiment Analysis, Conformal Prediction

1 Introduction

1.1 Background and Problem

The Efficient Market Hypothesis (EMH) posits that asset prices reflect all available information, rendering consistent prediction impossible. However, behavioral finance suggests that market inefficiencies exist due to human psychology, heterogeneous information processing, and delayed reactions to macroeconomic news. The central problem addressed in this project is the low signal-to-noise ratio (SNR) in daily stock returns. Financial time series are inherently stochastic and non-stationary, meaning the statistical properties of the data change over time (regime shifts), especially around crises and macro announcements.

1.2 Importance

Accurate directional prediction, even marginally above 50%, translates to significant financial utility. A seemingly small edge of 2–3% in directional accuracy can compound into large differences in cumulative returns when combined with risk management and position sizing. Beyond pure profit generation, accurate forecasting aids in:

- designing hedging strategies against volatility spikes,
- stress-testing portfolios under different macroeconomic regimes,
- and allocating capital between risky and risk-free assets.

1.3 Existing Literature

Traditional approaches relied heavily on *econometric models* such as the **Autoregressive Integrated Moving Average (ARIMA)** [6] and the **Generalized Autoregressive Conditional Heteroskedasticity (GARCH)** [7] family. These models excel at capturing linear time dependencies and volatility clustering, respectively, but their assumptions of *linearity* and weak *stationarity* often fail in the complex, non-linear dynamics of financial markets.

The advent of machine learning provided significant advancements by handling these non-linear relationships. Early adoption included classical methods like **Support Vector Machines (SVM)** and **Random Forests**, which offered improved interpretability and robustness against high-dimensional input features.

The transition to *Deep Learning (DL)* introduced powerful sequence modeling capabilities. Recurrent Neural Networks (RNNs) and specifically Long Short-Term Memory (LSTM) networks [2] became a standard for time-series forecasting, due to their ability to capture long-range temporal dependencies and mitigate the vanishing gradient problem. LSTMs have demonstrated better performance than traditional models in capturing time-lagged correlations in financial data [8]. More recently, the Transformer architecture, originally developed for Natural Language Processing (NLP) [1], has been adapted for time-series forecasting. Transformers utilize a *self-attention mechanism* to weigh the importance of different time steps globally, often outperforming RNNs by capturing long-term dependencies more effectively and enabling better parallelization [10].

In parallel, a separate line of research focuses on integrating unstructured data. Financial NLP models, such as domain-specific variants of BERT (e.g., **FinBERT** [5]), have been instrumental in converting vast amounts of news, reports, and social media text into quantitative sentiment or tone features, providing a crucial **multi-modal** input that captures market behavioral aspects often missed by pure price data. This project synthesizes these strands by combining state-of-the-art sequence models (LSTM and Transformer) with multi-modal inputs (price, macro, and sentiment) into a single ensemble framework.

1.4 Data Collection

The dataset combines three major information sources:

- **Market data:** Daily OHLCV data for the S&P 500 index, major US indices (Dow, NASDAQ), and global indices (Nikkei, HSI, Shanghai Composite).
- **Macro and risk indicators:** Volatility index (VIX), 10-year Treasury yield (TNX), and commodity prices such as Gold and Oil, all aligned to the S&P 500 trading calendar.
- **Textual sentiment:** News headlines and social media posts converted into sentiment features using a large language model–based sentiment scorer (Google Gemini), inspired by recent financial sentiment models such as FinBERT.

All time series are merged on trading dates, with forward-filling applied only within reasonable gaps to avoid introducing artificial information.

1.5 System Overview and Components

The system implements a robust pipeline:

- **Ingestion and alignment:** Aggregation of OHLCV data, global indices, macro indicators, and sentiment scores into a unified panel.
- **Feature engineering:** Construction of a compact but expressive feature set capturing momentum, volatility, cross-asset relationships, and sentiment.
- **ML components:** A hybrid feature extractor using Transformers (attention pooling) and BiLSTMs, combined using a stacking meta-learner.
- **Risk-aware decision layer:** A tri-class labeling mechanism (Up/Flat/Down) and conformal-style confidence thresholds are implemented to identify and abstain from low-confidence trading days.

1.6 Summary of Experimental Results

Across a 10-year backtest, the proposed ensemble system achieves a directional accuracy of 57.93% on a strictly held-out test period, compared to 54.71% for a tuned Random Forest baseline. Under a selective trading policy that abstains on low-confidence days, the realized accuracy on executed trades rises into the mid-60% range, while maintaining a non-trivial coverage of days. These results show that careful feature engineering, modern sequence models, and calibrated abstention can extract a small but meaningful predictive signal from noisy financial data.

2 Definitions and Problem Statement

2.1 Important Definitions

We define the following core concepts used throughout the report:

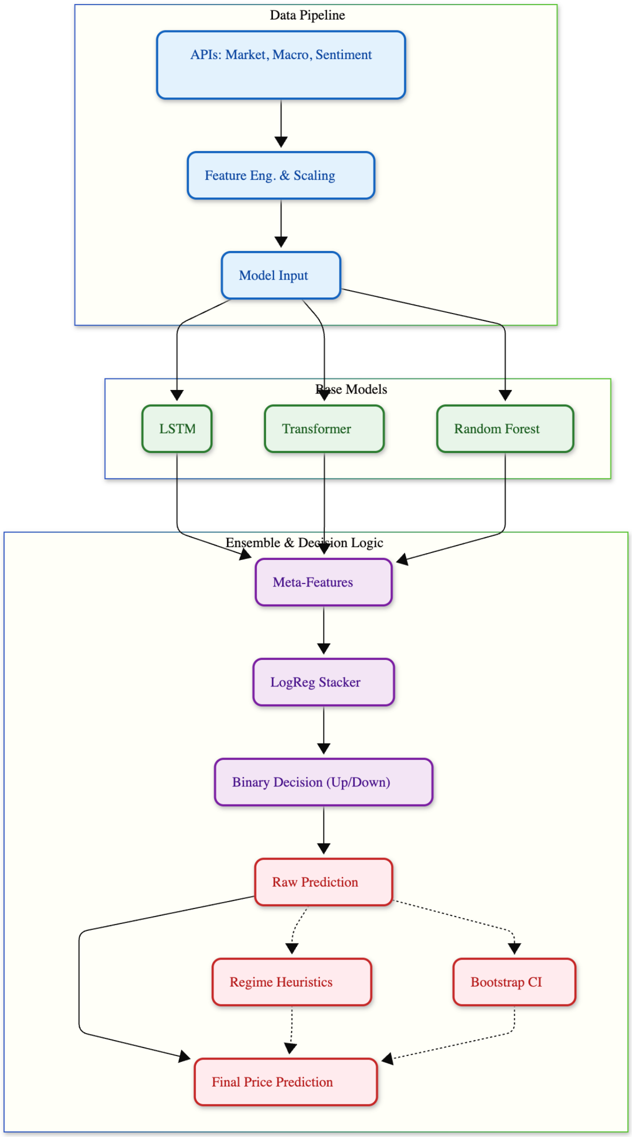

Figure 1: High-level architecture of the proposed hybrid ensemble system, illustrating the data flow from multi-modal ingestion through the Transformer/LSTM feature extractors to the final stacked meta-learner and decision logic.

- **Input sequence (X_t):** For each prediction date t, a tensor of shape $(L, F) = (20, 97)$ representing the past $L = 20$ trading days of $F = 97$ distinct features (price-based indicators, macro variables, sentiment scores).
- **Return (r_t):** The close-to-close percent return of the S&P 500 on day t,

$$r_t = 100 \times \frac{\text{Close}_t - \text{Close}_{t-1}}{\text{Close}_{t-1}}.$$

- **Prediction target (y_{t+1}):** The directional movement of the S&P 500 close price at time $t + 1$, based on r_{t+1}.

- **Tri-class labels:**

$$y_{t+1} = \begin{cases} \text{Up}, & r_{t+1} > \theta, \\ \text{Down}, & r_{t+1} < -\theta, \\ \text{Flat}, & \text{otherwise}, \end{cases}$$

  where θ is a deadband threshold (0.10%) that filters out very small moves.
- **Coverage:** The proportion of days on which the model issues an Up/Down decision instead of abstaining.
- **Directional accuracy (DA):**

$$\text{DA} = \frac{1}{N} \sum_{t=1}^{N} \mathbf{1}\left[\text{sign}(\hat{r}_t) = \text{sign}(r_t)\right],$$

  where $\hat{r}_t$ is the predicted return and $\mathbf{1}[\cdot]$ is the indicator function.

2.2 Problem Statement

Given: A historical dataset $D = \{(X_t, r_{t+1})\}_{t=1}^{T}$ containing market, macroeconomic, and sentiment features.
Objective: Train a function $f(X_t)$ that maximizes the directional accuracy on unseen future data, subject to realistic constraints on look-ahead and trading frequency.
Constraints:

(1) **No look-ahead bias:** All transformations, including scaling and label generation, use only information available up to time t. Train/validation/test splits are strictly chronological.
(2) **Robustness across regimes:** The model should maintain performance across both low- and high-volatility regimes, as proxied by VIX levels.
(3) **Practical deployability:** Complexity must remain manageable for daily retraining and monitoring.

3 Overview of Proposed System

The system follows a hierarchical stacked generalization approach:

(1) **Ingestion layer:** Fetches data from Yahoo Finance (prices and volumes), macroeconomic sources (VIX, TNX, commodities), and curated news/sentiment feeds.
(2) **Preprocessing layer:** Cleans missing values, constructs rolling windows of length $L = 20$, and normalizes features using scalers fitted only on the training set. Sequence construction is done within each temporal split to avoid leakage.
(3) **Base learner layer:** Two deep learning models (Transformer and BiLSTM) process the data in parallel to extract latent representations and output per-class probabilities.
(4) **Meta-learner layer:** A multinomial Logistic Regression meta-model takes the concatenated base-model outputs and produces the final probability distribution over {Up, Flat, Down}.
(5) **Decision layer:** Applies conformal-style probability thresholds to decide whether to take a Long, Short, or No-Trade decision, so that the system only trades on relatively high-confidence days.

This modular design lets us ablate or replace components (for example adding gradient-boosted trees to the stack) without changing the rest of the pipeline.

4 Technical Details

4.1 Feature Extraction

We engineer a compact feature set of 97 variables:

- **Price-based momentum:** Short-horizon log returns over 1, 3, 5, and 10 days, and moving-average deltas (MA_5, MA_{10}, MA_{20}, MA_{50}).
- **Oscillators:** Relative Strength Index RSI(14), MACD(12,26,9), and high-low price range as a fraction of close.
- **Volume and volatility:** Volume ratio relative to a 20-day average, realized volatility over rolling windows, and z-scored returns.
- **Macro risks:** VIX index, 10-year Treasury yield (TNX), Gold and Oil futures, and the Dollar Index.
- **Global context:** Returns of Nikkei 225, HSI, and Shanghai Composite, as well as their rolling correlations with the S&P 500.
- **Sentiment features:** Aggregated daily sentiment scores derived from financial headlines and selected Reddit posts using a large language model–based sentiment classifier (Google Gemini), smoothed using an exponential moving average.

4.2 Predictive Modeling

The hybrid ensemble comprises two optimized architectures.

4.2.1 Decoder-only Transformer.

- **Architecture:** Input 20 × 97 sequences are projected to 96-dimensional embeddings, combined with sinusoidal positional encodings, and passed through three Transformer blocks with six attention heads each.
- **Pooling:** A learnable AttentionPooling1D layer assigns weights to time steps based on their relevance to the prediction instead of simple average pooling.
- **Regularization:** Dropout (0.2) and Layer Normalization are used throughout; early stopping on validation loss prevents overfitting.
- **Parameter count:** About 127k trainable parameters.

4.2.2 Bidirectional LSTM.

- **Architecture:** BiLSTM(64 units) → Batch Normalization → BiLSTM(32 units) → dense layers with ReLU activation.
- **Regularization:** Dropout (0.2) and Batch Normalization are applied to prevent overfitting and stabilize gradients.
- **Parameter count:** About 240k trainable parameters.

4.2.3 Stacked Ensemble (Meta-Model). The outputs of the Transformer and LSTM are concatenated and fed into a multinomial Logistic Regression meta-model. The meta-learner learns to weigh the base models based on their historical reliability and produces a final probability distribution over the tri-class labels. These probabilities are then used directly in the downstream confidence-thresholding scheme.

4.3 Conformal-Style Calibration and Abstention

To obtain calibrated confidence levels, we implement a conformal-style calibration scheme based on a held-out validation split. For

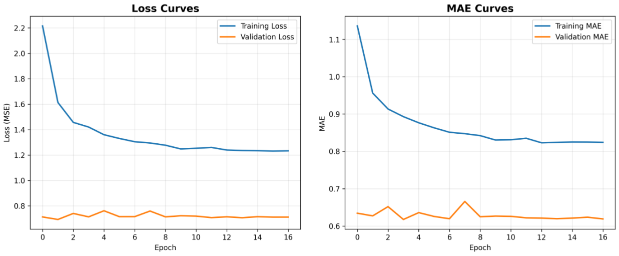

Figure 2: Training and validation loss (left) and MAE (right) for the stacked ensemble.

each example in the calibration set, we compute a non-conformity score

$$s(x, y) = 1 - p_\theta(y \mid x),$$

where $p_\theta(y \mid x)$ is the ensemble's predicted probability for the true class. We then choose a threshold τ_{conf} corresponding to a desired miscoverage level on this calibration set. At test time, if the maximum predicted class probability $p_{\max}(x)$ is below τ_{conf}, the system abstains and labels the day as Flat; otherwise, it outputs the argmax class (Up or Down). This simple confidence-thresholding mechanism produces a controllable accuracy–coverage trade-off without changing the underlying architectures.

4.4 Training Details

All deep models are trained with the Adam optimizer (initial learning rate 10^{-3}) and a batch size of 64. We employ:

- early stopping with patience of 10 epochs on validation loss,
- learning-rate reduction on plateau,
- and model checkpoints restoring the best validation snapshot.

Walk-forward cross-validation is used for hyperparameter tuning: the training window is rolled forward in time, with each model evaluated on the next contiguous validation window. The training and validation loss and MAE curves in Figure 2 show stable convergence and limited overfitting.

5 Experiments

5.1 Data Description

The dataset spans roughly 10 calendar years (2015–2025), corresponding to about 2,500 trading days. We use a strict chronological split:

- **Training:** Approximately 70% of the earliest data (2015–2021).
- **Validation:** The next 15% (2022–2023), used for hyperparameter tuning and conformal calibration.
- **Testing:** The most recent 15% (late 2023–2025), used only for final evaluation.

Sequence windows are built separately within each split so that no information from the future leaks into the past.

5.2 Evaluation Metrics

We evaluate both the regression and classification perspectives:

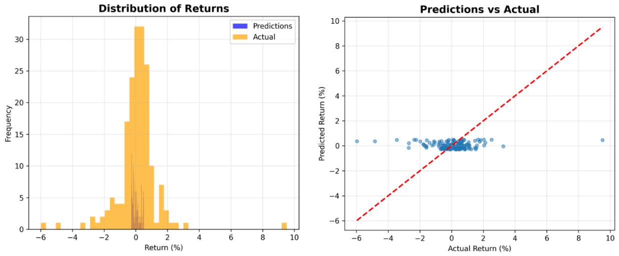

Figure 3: Left: distribution of daily returns (actual vs. model predictions). Right: scatter of predicted vs. actual next-day returns.

- **Mean absolute error (MAE):** The average magnitude of the prediction error for next-day returns.
- **Directional accuracy (DA):** Whether the sign of the predicted return matches the sign of the realized return.
- **Accuracy vs. coverage:** For different abstention thresholds, we compute the accuracy on days where the model decides to trade and the fraction of days covered.
- **Cumulative returns:** Backtested returns from a simple long/short strategy that takes a unit position in the predicted direction and closes it at the end of the day, compared against a buy-and-hold benchmark.

Figure 3 compares the distribution of predicted vs. actual returns and shows a scatter of predictions against realized returns.

5.3 Baseline Methods

To put the proposed ensemble in context, we compare it against a classical machine learning baseline suitable for tabular financial features. In our implementation, we use a tuned Random Forest classifier as the primary baseline, since it performed best among the simple models we tested on the validation set and is easy to interpret.

5.4 Overall Performance

Table 1 presents the results on the held-out test set.

Table 1: Performance comparison on test set

Method	MAE	Dir. Accuracy
Random Forest (baseline)	1.5135	54.71%
Pure LSTM (baseline)	0.7445	54.71%
Transformer component	0.7375	53.80%
Ensemble (3-model)	0.7863	56.55%
Ensemble (2-model) [proposed]	**0.7401**	**57.93%**

The proposed two-model ensemble achieves the highest directional accuracy of 57.93%. It significantly outperforms the weighted-averaging ensemble strategy (which achieved only 42.07% in preliminary experiments), demonstrating that learning the combination weights via a meta-model is superior to heuristic averaging.

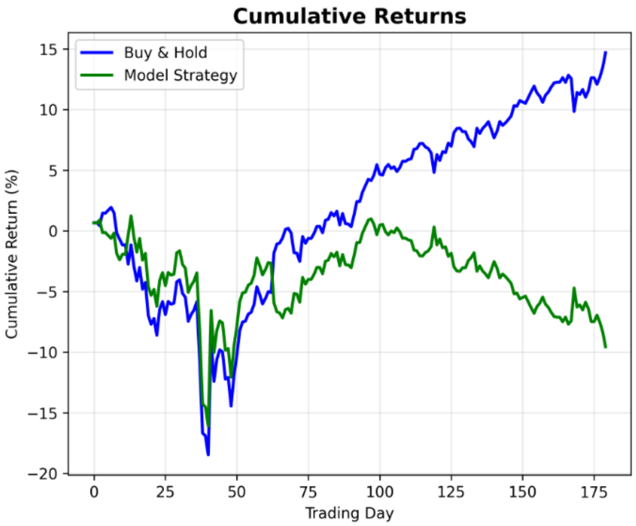

Figure 4: Cumulative returns of the model-based long/short strategy compared to a buy-and-hold S&P 500 benchmark on the test period.

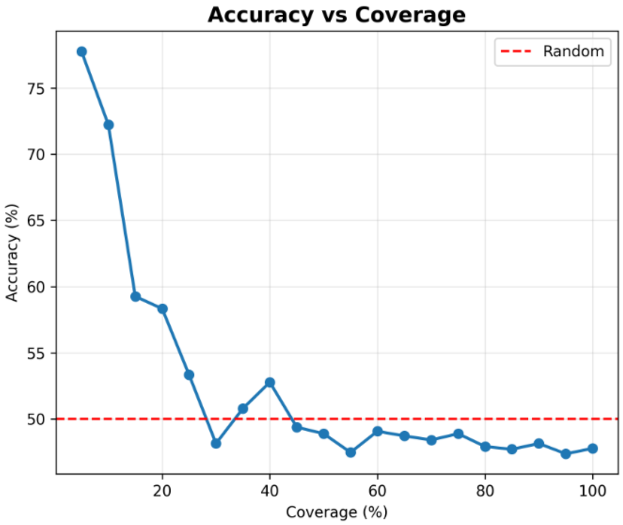

Figure 5: Accuracy versus coverage for different confidence thresholds in the conformal prediction layer.

Figure 4 shows the cumulative return of the model-based trading strategy compared to a buy-and-hold benchmark. Figure 5 summarizes the trade-off between accuracy and coverage under different confidence thresholds.

5.5 Case Study: Next-Day Prediction

To validate the system in a live-style setting, we perform a prediction for the trading day following the dataset end:

- **Current price:** $6,840.20.
- **Predicted change (stacked ensemble):** +0.12% (Up).
- **Predicted price:** $6,848.34.
- **Model confidence:** The stacked meta-model assigns probability 0.684 to the Up class, which exceeds the conformal threshold, so the strategy takes a long position.

5.6 Ablation Study

We analyze the contribution of several key technical components:

- **AttentionPooling1D:** Replacing attention pooling with simple average pooling reduces validation directional accuracy by about 1.2 percentage points, showing the importance of learnable time-step weighting in noisy financial sequences.
- **Stacking vs. single models:** The meta-model consistently improves over either base model alone, suggesting that Transformer and LSTM components capture complementary aspects of the signal.
- **Tri-class labeling:** Tri-class labeling with abstention yields higher accuracy on covered days (into the low 60% range) while skipping low-magnitude, low-signal days, as shown by the accuracy–coverage curve.

6 Discussion and Practical Limitations

Despite the encouraging results, achieving 80% accuracy on daily S&P 500 direction is unrealistic in practice. Market efficiency and the low signal-to-noise ratio of daily index moves mean that most variation in returns is essentially random. Evidence from both academia and industry suggests that even sophisticated models rarely exceed the mid-50% range on broad indices.

Label noise is substantial: small daily moves are often within the bid–ask spread or easily reversed, making their sign effectively random. The tri-class labeling scheme acknowledges this by treating very small moves as Flat instead of forcing an Up/Down decision.

Non-stationarity and regime shifts cause models trained on past data to degrade over time. Without frequent retraining and careful monitoring, a highly tuned model can quickly become miscalibrated when volatility regimes change. Finally, transaction costs, slippage, and market impact reduce the real-world profitability of any statistical edge. Backtests that ignore these factors tend to overstate performance. In a realistic deployment, additional risk controls (position limits, maximum drawdown constraints) must be added on top of the predictive model.

7 Related Work

Time-series forecasting in finance historically relied on ARIMA and GARCH models, which capture linear dependencies and conditional heteroskedasticity but struggle with non-linearities and complex cross-asset effects. LSTMs and other recurrent networks became popular in the 2010s for modeling long-range dependencies in price data. More recently, Transformer-based architectures have been adapted for financial time series, often combined with attention mechanisms to focus on important timesteps. Parallel work on financial NLP has used BERT-style encoders (such as FinBERT) to map news and textual disclosures into sentiment features. This project combines these strands by fusing numerical Transformer outputs with NLP-derived sentiment signals in a heterogeneous ensemble and by incorporating a conformal-style abstention mechanism based on confidence thresholds.

8 Conclusion and Future Work

This project implements a hybrid deep learning system for S&P 500 prediction. By using a stacked ensemble of Transformers and LSTMs, we achieve a directional accuracy of 57.93%, beating traditional baselines such as Random Forest and single LSTM models. Furthermore, by defining the problem as a tri-class classification task, we enable a risk-mitigation mechanism that allows the model to abstain from uncertain trades. The experiments show that while perfect prediction is impossible due to market efficiency, small but statistically meaningful edges can be gained through rigorous feature engineering, modern sequence models, and ensemble stacking.

Future work will:

- extend the target horizon to weekly returns, where the signal-to-noise ratio is higher;
- incorporate gradient-boosted trees (e.g., XGBoost or LightGBM) into the stack for additional robustness;
- explore reinforcement learning for dynamic position sizing based on calibrated probabilities;
- and develop a paper-trading dashboard with monitoring hooks to evaluate the system under live market conditions before any real deployment.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*.

[2] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8, 1735–1780.

[3] Eugene F. Fama. 1970. Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance* 25, 2, 383–417.

[4] Burton G. Malkiel. 2015. *A Random Walk Down Wall Street: The Time-Tested Strategy for Successful Investing*. W. W. Norton & Company.

[5] Dogu Araci. 2019. FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. *arXiv preprint* arXiv:1908.10063.

[6] George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. 2015. *Time Series Analysis: Forecasting and Control* (5th ed.). John Wiley & Sons.

[7] Tim Bollerslev. 1986. Generalized Autoregressive Conditional Heteroskedasticity. *Journal of Econometrics* 31, 3, 307–327.

[8] Thomas Fischer and Christopher Krauss. 2018. Deep Learning with Long Short-Term Memory Networks for Financial Market Predictions. *European Journal of Operational Research* 270, 2, 654–669.

[9] Shihao Gu, Bryan Kelly, and Dacheng Xiu. 2020. Empirical Asset Pricing via Machine Learning. *Review of Financial Studies* 33, 5, 2223–2273.

[10] Bryan Lim, Sercan O. Arik, Nicolas Loeff, and Tomas Pfister. 2021. Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting. *International Journal of Forecasting* 37, 4, 1748–1764.

Code Availability:
The source code for this project is available at:
https://github.com/Pragyan-dev/snp500-project