Predicting S&P 500 Index Closing Prices: A Multi-Model Machine Learning Approach

Swarnima Joshi, Krutarth Chaudhari, Yu Xia

I. Data Collection and Preprocessing

Historical market data for the S&P 500 index and supplementary indicators (VIX, 10-Year Treasury Yield, US Dollar Index) were downloaded using the Yahoo Finance API from January 2000 through April 21, 2025 that included information about Opening price, High price, Low price, Closing price and Trading volume. To enhance the predictive power of our models, we supplemented the S&P 500 data with additional market indicators like VIX, `USD_INDEX` and 10-Year Treasury Yield. The data cleaning process involved identifying missing values, dropping rows with NaN values, performing inner joins to ensure temporal alignment, and standardizing column names with clear prefixes (e.g., `'SPX_Close'`, `'VIX_Close'`). The cleaned dataset was verified for completeness and saved to a CSV file for subsequent analysis, making it computationally efficient by not repeating the data processing step everytime we ran the code.

II. Feature Engineering

We engineered a total of 34 features to capture market behavior across multiple dimensions:

- **Price-based Features:** Lagged closing prices (`Close_lag_1`, `Close_lag_2`, `Close_lag_3`), moving averages (`MA5`, `MA10`, `MA20`, `MA50`), daily price change (`Price_Change`), close-to-open ratio (`Close_Open_Ratio`), and high-low range (`High_Low_Range`).
- **Return-based Features:** Daily returns (`Returns`), logarithmic returns (`Log_Returns`), and multi-period returns (`Returns_5d`, `Returns_20d`).
- **Volatility Measures:** 10-day rolling standard deviation (`Volatility`) and Average True Range (`ATR`).
- **Volume Indicators:** Raw trading volume (`Volume`), 5-day volume moving average (`Volume_MA5`), and volume ratio (`Volume_Ratio`).
- **Technical Indicators:** Relative Strength Index (`RSI`), Moving Average Convergence Divergence (`MACD`), and Average True Range (`ATR`).
  RSI is calculated over a 14-day period by computing daily price changes, separating gains and losses, calculating rolling averages, and deriving the relative strength (RS = Avg Gain / Avg Loss). It is then transformed to a 0–100 scale using: $RSI = 100 - \left(\frac{100}{1+RS}\right)$.
  MACD is derived from the difference between the 12-day (fast) and 26-day (slow) exponential moving averages (EMAs). A 9-day EMA of the MACD line acts as the signal line, while the histogram represents the difference between the MACD line and the signal line.
  ATR is a volatility measure calculated over 14 days using the True Range (TR), defined as the maximum of: high–low, $|\text{High} - \text{Prev Close}|$, and $|\text{Low} - \text{Prev Close}|$. ATR is typically computed as the EMA of the TR.
- **Market Indicators:** VIX (`VIX`, `VIX_MA5`, `VIX_Change`), 10-Year Treasury Yield (`T10Y`, `T10Y_MA5`, `T10Y_Change`), and US Dollar Index (`USD`, `USD_MA5`, `USD_Change`), including their moving averages and rate-of-change derivatives.

Missing values were handled through forward filling followed by backward filling to preserve temporal patterns. Features were standardized using StandardScaler to ensure equal contribution to model training.

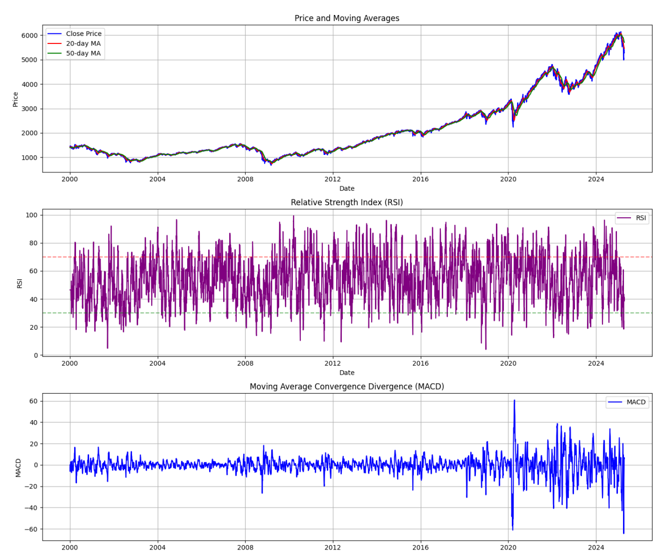

Fig. 1: Technical Analysis

III. Train-Test Split

To maintain temporal integrity crucial for time series forecasting, we implemented a chronological split:

- **Training Set:** Data from January 2000 to October 23, 2024 (6,235 days).
- **Testing Set:** Data from October 24, 2024 to April 17, 2025 (120 days).

The closing price range was $676.53 - 5,864.67$ (mean: $2,083.80$) for the training set and $4,982.77 - 6,144.15$

(mean: $5,854.82$) for the testing set. This difference presented a challenging but realistic scenario for evaluating model generalization.

IV. Modeling Methodology

We developed an ensemble approach combining three distinct models:

Random Forest Regressor: After experimenting with tree counts (100→200→300) and depths (10→15), we settled on 300 trees with `max_depth=15`, `min_samples_split=5`, and `min_samples_leaf=2`. This model effectively captured non-linear relationships and provided feature importance rankings.

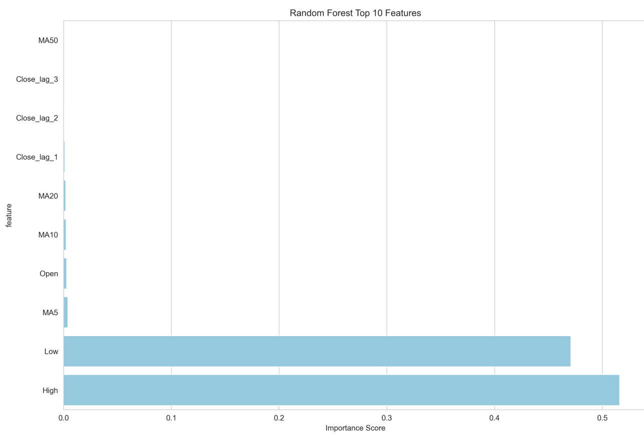

Fig. 2: Random Forest Feature Importance

XGBoost Regressor: Initial tests with 100–200 estimators and 0.1–0.2 learning rates showed that 300 estimators with a slower learning rate (0.05) provided better stability. Final parameters included `max_depth=8`, `subsample=0.8`, `colsample_bytree=0.8`, `min_child_weight=1`, and `gamma=0.1`.

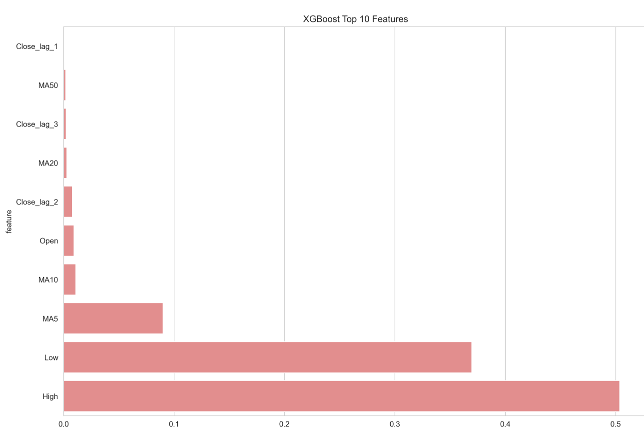

Fig. 3: XGBoost Feature Importance

SARIMA: After testing various parameter combinations, we selected `order=(2,1,2)` for the non-seasonal component and `seasonal_order=(1,1,1,5)` for the seasonal component, with stationarity and invertibility constraints disabled.

These were the 10 most important features for RF and XGBoost: `High`, `Low`, `MA5`, `MA10`, `Open`, `Close_lag_2`, `MA20`, `Close_lag_3`, `MA50`, `Close_lag_1`.

A. Ensemble Method

We implemented a performance-based weighted ensemble where weights were calculated as the inverse of each model's mean absolute error, normalized to sum to 1. The final model weights were: Random Forest (44.07%), XGBoost (29.88%), and SARIMA (26.06%). This approach allowed the ensemble to adapt to changing market conditions by favoring better-performing models.

$$\text{weight}_{\text{model}} = \frac{1/\text{MAE}_{\text{model}}}{\sum(1/\text{MAE}_{\text{all models}})}$$

$$\text{prediction}_{\text{ensemble}} = \sum(\text{weight}_{\text{model}} \times \text{prediction}_{\text{model}})$$

B. Confidence Interval Estimation

To quantify prediction uncertainty, we computed 95% confidence intervals for each forecast by collecting predictions from all models on each date and calculating their standard deviation (σ). The interval was defined as:

$$[\text{ensemble\_prediction} \pm 1.96 \times \sigma]$$

This reflects model disagreement: narrow intervals suggest high certainty, while wider ones indicate greater uncertainty, especially in volatile market conditions.

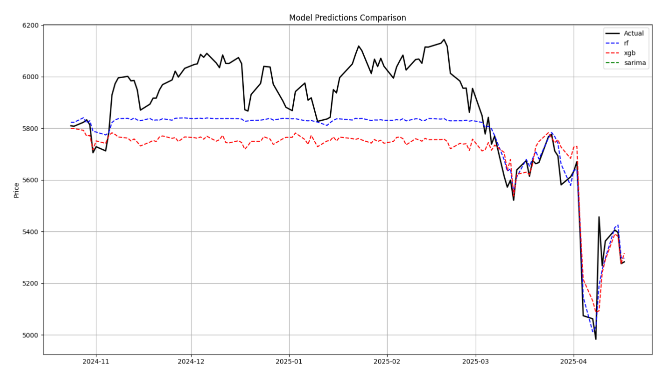

Fig. 4: Model Comparison

C. Future Prediction Methodology

For predicting future dates (April 22–25, 2025), we used a rolling approach starting from data through April 21. For each forecast day, we generated features using previous predictions for lagged values, updated technical indicators (RSI, MACD, ATR), and kept other market indicators fixed at their last known values. Each model in the ensemble produced predictions, which were combined into a weighted ensemble forecast. This prediction was then added to the dataset to generate features for the next day. Confidence intervals (95%) were computed using the standard deviation of individual model predictions: (prediction $\pm 1.96\sigma$).

V. Results and Analysis

A. Model Performance

On the test set (Oct 24, 2024 - Apr 17, 2025), the Random Forest model demonstrated superior performance individually, achieving a Mean Absolute Error (MAE) of 120.02, Root Mean Squared Error (RMSE) of 149.69, and an R-squared (R^2) value of 0.6175. The XGBoost model showed higher error rates (MAE=177.49, RMSE=211.21, R^2=0.2385). The weighted ensemble achieved intermediate performance (MAE=140.78, RMSE=173.56, R^2=0.4858), indicating it did not outperform the best individual model in this instance.

Feature importance analysis confirmed that lagged prices, short-term moving averages (MA5, MA10), volatility measures (ATR), and technical indicators were the most influential predictors for the tree-based models.

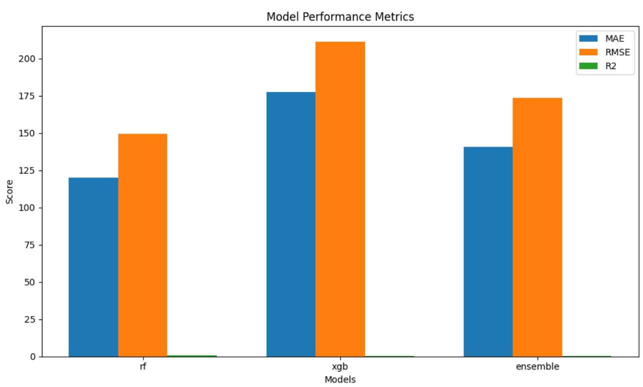

Fig. 5: Ensemble Model Residuals

B. Prediction vs. Actual (April 22-25, 2025)

The model predicted a modest upward trend for the four-day forecast period, with closing prices expected to increase from 5299.00 on April 22 to 5302.68 on April 25 (a total increase of +0.07%). However, comparison with the actual closing prices revealed significant discrepancies, as shown in Table I.

TABLE I: Comparison of Predicted vs. Actual Closing Prices

Date	Predicted	Actual	Difference	Error (%)
2025-04-22	5299.00	5287.76	+11.24	+0.21%
2025-04-23	5301.19	5375.86	−74.67	−1.39%
2025-04-24	5301.97	5484.77	−182.80	−3.33%
2025-04-25	5302.68	5525.21	−222.53	−4.03%

C. Limitations and Future Improvements

The model failed to capture the significant market rally that occurred during the forecast period (+4.49% actual vs +0.07% predicted) and exhibited increasing error as the forecast horizon extended. Key limitations and potential areas for improvement include:

- **Trend Detection Weakness:** The model lacked sensitivity to emerging momentum. Future work should incorporate enhanced momentum indicators (e.g., Rate of Change, Money Flow Index) and potentially incorporate signals from options market data (e.g., put/call ratios) which often reflect sentiment and anticipated volatility.
- **Volatility Underestimation:** GARCH models and adaptive confidence intervals would provide more realistic uncertainty estimates during volatile periods.
- **Error Propagation:** The increasing error magnitude highlights the compounding effect in multi-step forecasting. Developing error correction mechanisms (e.g., using the previous day's error to adjust the current forecast) or retraining the model more frequently within the prediction window might mitigate this issue.
- **Ensemble Optimization:** The ensemble underperformed the best individual model (Random Forest). Exploring advanced ensemble techniques like stacking (where a meta-model learns to combine base model predictions) or Bayesian model averaging could potentially yield better results by more effectively leveraging the strengths of each constituent model.
- **Dynamic Market Adaptation:** Financial markets exhibit different regimes (e.g., trending, mean-reverting, high/low volatility). Implementing regime-switching models (e.g., Markov-switching models) that adapt their parameters or select different prediction strategies based on the detected market state could improve performance across diverse market conditions.

References

[1] Wall Street Journal Market Data. S&P 500 Historical Prices. Available: https://www.wsj.com/market-data/quotes/index/SPX/historical-prices [Accessed: April 21, 2025]

[2] Yahoo Finance API (accessed via `yfinance` Python library). Available: https://pypi.org/project/yfinance/