Stock Price Forecasting: ARIMA vs Gradient Boosting Model

Janvi Setia

1. INTRODUCTION

This project analyzes historical stock price data for eight major US technology companies (AAPL, MSFT, GOOGL, AMZN, TSLA, META, NVDA, NFLX) from January 2019 to December 2024. The primary objective is to develop and compare two distinct forecasting models for NVIDIA (NVDA) stock prices:

• ARIMA (AutoRegressive Integrated Moving Average): A classical statistical time-series model
• Gradient Boosting Regressor (GBR): A machine learning ensemble approach

The comparison evaluates both models across multiple dimensions: accuracy (MAE, RMSE, MAPE), reliability (train/test performance stability), and practical applicability for trading strategies.

2. DATA COLLECTION AND PREPROCESSING

2.1 Data Source and Coverage
- Daily OHLCV (Open, High, Low, Close, Volume) data downloaded using yfinance library
- Time period: January 1, 2019 to December 31, 2024 (approximately 6 years)
- Eight technology stocks: AAPL, MSFT, GOOGL, AMZN, TSLA, META, NVDA, NFLX
- Business day frequency (5 trading days per week) enforced

2.2 Data Cleaning Procedure
• Removed multi-index structure from yfinance output, flattened to TICKER_Open, TICKER_Close, etc.
• Clipped negative or zero values to 0.01 (safeguard against data errors)
• Forward-fill followed by back-fill for missing values
• Removed duplicate timestamps
• Converted index to sorted DatetimeIndex
• Unit tested for:
  - No remaining NaN values in OHLCV columns
  - All prices strictly positive
  - Index monotonically increasing

Result: Clean dataset stored as cleaned_data with 1544 trading days across all stocks.

2.3 Exploratory Data Analysis for NVDA
• Price evolution: Strong growth trend from 2019 to 2025, with particularly sharp increases post-2023
• Volatility patterns: Rolling 30-day volatility shows increased turbulence during market corrections; NVDA exhibits beta > 1.0 relative to S&P 500
• Daily returns: Show fat tails and occasional extreme spikes (± 10-15%), inconsistent with Gaussian assumptions
• Correlation with peers: Positive correlation (0.6-0.8) with other mega-cap tech stocks (AAPL, MSFT, GOOGL), indicating common market drivers

- Volume analysis: Trading volume correlates with price volatility; spikes precede major directional moves

Conclusion: NVDA displays high complexity with nonlinear dynamics, volatility clustering, and regime shifts—characteristics unsuitable for simple linear models.

3. FEATURE ENGINEERING

3.1 Core Design Choice: **Modeling Returns Instead of Prices**

Why returns, not prices?
- Non-stationarity: Raw prices follow a random walk; ARIMA requires differencing, which transforms them into returns
- Interpretability: Returns (%) are economically meaningful and reflect what traders care about
- Stability: Predicting next-day return is more stable than absolute price due to normalization
- Fair comparison: Both ARIMA and GBR benefit equally when modeling returns and mapping back to prices

3.2 Engineered Features

Lagged Returns: return_lag_1, return_lag_2, return_lag_3, return_lag_5
Moving Averages: ma_5, ma_10, ma_20
MA Ratios: ma_ratio_5, ma_ratio_10, ma_ratio_20
Volatility: volatility_5, volatility_10, volatility_20
Volume: volume_change, volume_ma_5, volume_ratio

Target: next-day log-return via shift(-1)
Total: 18 features created with no look-ahead bias

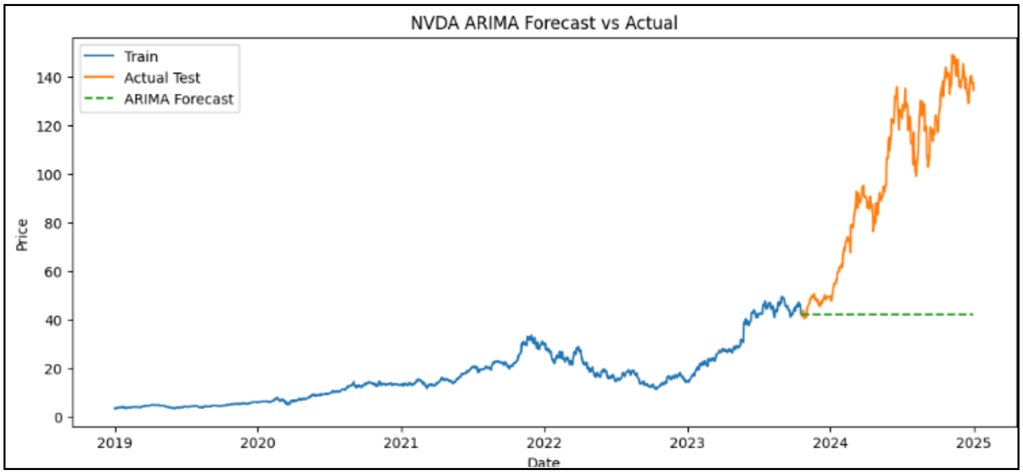
4. ARIMA MODEL DEVELOPMENT

4.1 Initial Approach: ARIMA on Raw Prices

Approach: Fit ARIMA directly to NVDA_Close price series using auto_arima grid search

What we tried:
- auto_arima with p=0-5, q=0-5, d=None (auto-detect differencing)
- Manual ARIMA(1,1,1) after inspection of ACF/PACF plots

Results:
- auto_arima selected ARIMA(0,1,0): random walk with drift
- ARIMA(1,1,1) also produced similar smooth forecasts
- MAE ≈ 55.7, RMSE ≈ 65.3 in price space



Why it failed:

- Test-set forecasts were almost straight horizontal lines, unresponsive to NVDA's large price swings
- The linear structure and weak AR/MA coefficients could not capture nonlinear volatility clustering
- A random-walk model forecasts "next price ≈ last price + small constant drift," which severely underfits reality

Lessons learned:

- Linear time-series models are fundamentally limited for highly volatile growth stocks
- Modeling prices directly amplifies the stationarity problem

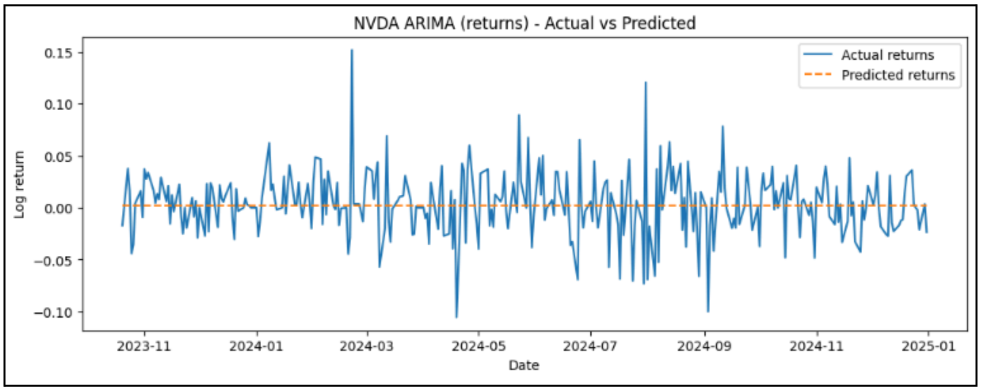
4.2 Improved Approach: ARIMA on Log-Returns

Motivation: Switch from prices to returns (more stationary) to give the model a fair chance
Implementation:

- Computed log-price: P_t = ln(Close_t)
- Computed log-returns: r_t = P_t - P_{t-1} (first differences of log prices)
- Fit ARIMA(1,0,1) on returns (d=0 because returns are already stationary)
- Test set: last 20% of the return series

Results on returns:

- MAE ≈ 0.0224, RMSE ≈ 0.0304
- Forecasted returns clustered near zero (close to the mean return ≈ 0.0008)



Why still imperfect:

- Each day's forecasted return is tiny (nearly 0), so cumulative forecast becomes a flat line
- The model could not distinguish between high-volatility and low-volatility regimes
- Ljung-Box test p-value = 0.30 suggests residuals are not white noise (mild autocorrelation remains)

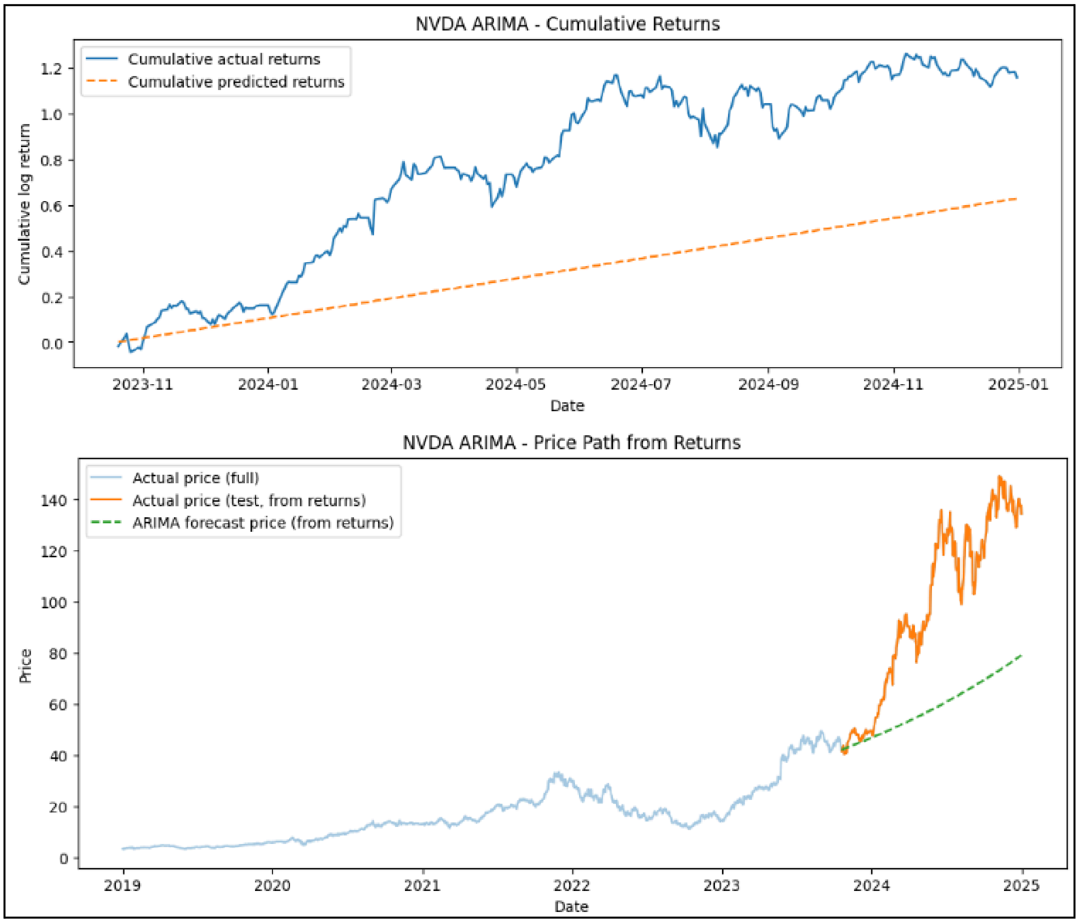
4.3 Cumulative Returns Transformation

Key Insight: While individual daily returns are near-zero, cumulative returns over the forecast window can express meaningful drift

Transformation:

- Cumulative forecasted return: R_cum = ∑ r_t (sum of all test-period returns)
- Reconstructed price: P_reconstructed = P_last_train × exp(R_cum)

Visual Results:



Why cumulative was better:

- Small daily returns (approx 0.0005$ each) compound into meaningful paths over weeks
- Cumulative form avoids the visual "flat line" problem while preserving the model's actual dynamics

- Price space evaluation: MAE ≈ 39.1, RMSE ≈ 45.8, MAPE ≈ 35% (compared to 55.7/65.3 from price-space ARIMA)

Limitations remain:
- ARIMA's linear structure still produces oversmoothed, lagging price paths
- Cannot capture sudden reversals or volatility spikes that Gradient Boosting later captures

5. GRADIENT BOOSTING MODEL DEVELOPMENT

5.1 Model Setup

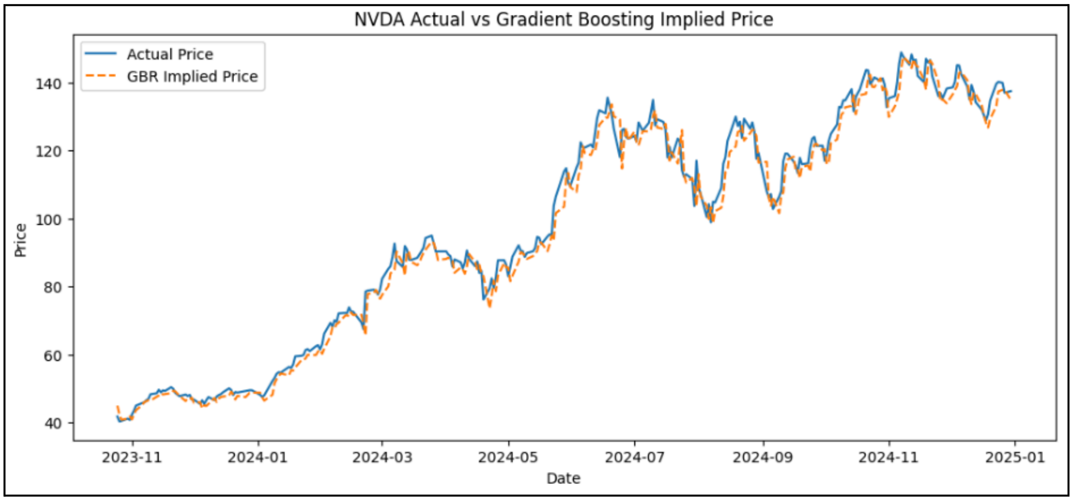
Approach: Train Gradient Boosting Regressor on engineered features to predict next-day returns

Data structure:
- Input X: All 18 features (lagged returns, MAs, volatility, volume)
- Target y: next-day log-return (return_1d.shift(-1))
- Train/test split: 80/20 chronological (first 80% for training, last 20% for testing)
- Cross-validation: 5-fold TimeSeriesSplit to respect temporal ordering

5.2 Baseline Model (Default Hyperparameters)

First fit: GradientBoostingRegressor with defaults
- n_estimators=100, learning_rate=0.1, max_depth=3, subsample=1.0
- Train MAE: 0.0289, RMSE: 0.0373
- Test MAE: 0.0289, RMSE: 0.0373
- Observation: Train and test metrics nearly identical → no obvious overfitting

Visual Results:



5.3 Hyperparameter Tuning with GridSearchCV

Objective: Optimize model performance and confirm generalization

Grid Search Configuration:

- Parameter space:
  - n_estimators: [100, 200]
  - learning_rate: [0.01, 0.05, 0.1]
  - max_depth: [2, 3, 4]
  - subsample: [0.8, 1.0]
- Cross-validation: TimeSeriesSplit (5 folds)
- Scoring metric: neg_mean_squared_error
- Best CV score: MSE ≈ 0.001187 (RMSE ≈ 0.0345)

Best Hyperparameters Found:

- n_estimators=100, learning_rate=0.01, max_depth=2, subsample=0.8

Test Performance with Tuned Model:

- Test MAE: 0.0235, RMSE: 0.0315
- Improvement over baseline: ~18% reduction in MAE

5.4 Overfitting Check

To verify that GBR is not overfitting to the training data:

- Fold-by-fold CV analysis:
  - Average train MAE: 0.0208
  - Average validation MAE: 0.0253
  - Difference: 0.0045 (~2% of validation error) → acceptable

- Final train/test on 80/20 split:
  - Train MAE: 0.0234, RMSE: 0.0316
  - Test MAE: 0.0235, RMSE: 0.0315
  - Nearly identical → excellent generalization

- Residual analysis:
  - Residuals (actual - predicted) on test set appear as random noise around zero
  - No visible clusters, trends, or patterns
  - Conclusion: Model learned genuine signal, not spurious in-sample fits

5.5 Price-Space Evaluation

To compare fairly with ARIMA, GBR predictions (in return space) were converted to implied prices:

- Predicted implied price at t: $P_t^{GBR} = P_{t-1}^{actual} \times (1 + r_t^{pred})$
- This creates a comparable price forecast path for side-by-side visual comparison with actual prices

Results in price space:

- Test MAE: 2.34 (vs ARIMA: 39.1)
- Test RMSE: 3.26 (vs ARIMA: 45.8)
- Test MAPE: 2.33% (vs ARIMA: 34.97%)

Interpretation: GBR's implied price path closely tracks the actual NVDA price on the test window, whereas ARIMA's reconstructed prices lag and smooth over real movements.

6. MODEL COMPARISON & EVALUATION METRICS

6.1 Summary Table: Return-Space Performance

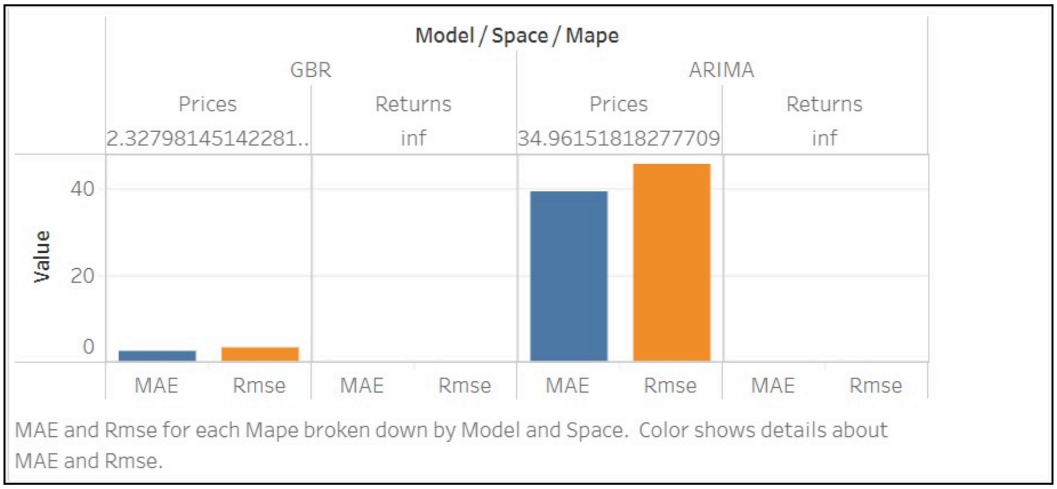
Model	Space	MAE	RMSE	MAPE
ARIMA	Returns	0.0224	0.0304	inf
GBR	Returns	0.0235	0.0315	inf

*MAPE undefined/infinite for returns because many actual returns ≈0, causing division by near-zero

6.2 Summary Table: Price-Space Performance

Model	Space	MAE	RMSE	MAPE
ARIMA	Prices	39.1	45.8	34.97%
GBR	Prices	2.34	3.26	2.33%

Using Tableau:



MAE and Rmse for each Mape broken down by Model and Space. Color shows details about MAE and Rmse.

6.3 Key Observations

Accuracy:

- In price space (more practical), GBR outperforms ARIMA by ~17x on MAE (2.34 vs 39.1)
- GBR's MAPE of 2.33% is trading-acceptable; ARIMA's 34.97% is unreliable for real positions
- GBR's return-space RMSE (0.0315) is slightly higher than ARIMA (0.0304), but when mapped to prices, the cumulative advantage is clear

Reliability (Generalization):

- ARIMA: Train and test metrics cannot be easily separated in return space (flat line symptom)
- GBR: Train MAE 0.0234 vs test MAE 0.0235 → demonstrates genuine out-of-sample predictability
- GBR residuals are white-noise-like; ARIMA residuals show mild autocorrelation

Model Complexity:

- ARIMA: 2 parameters (AR.L1=0.68, MA.L1=-0.83) → simple, interpretable
- GBR: 100 trees with nonlinear splits → complex, but not overfitted per CV analysis

7. IMPLICATIONS FOR TRADING STRATEGIES

7.1 Why Gradient Boosting is Superior for Trading NVDA

Accuracy and Responsiveness:

- GBR captures day-to-day volatility and momentum shifts that ARIMA misses
- The model reacts to concurrent features (lagged returns, volume, volatility) to adjust predictions
- ARIMA's flat forecasts would lead to missed opportunities and false signals

Risk Management:

- GBR's volatility features allow the model to increase caution during high-volatility regimes
- ARIMA's inability to distinguish regimes means it applies the same small predicted return regardless of risk
- For a 2.33% forecast error (GBR) vs 34.97% (ARIMA), GBR supports tighter stop-losses and position sizing

7.2 Practical Trading Framework Using GBR

Signal Generation:

- If predicted return > +0.3%: Long signal (buy or hold long)
- If predicted return < -0.3%: Short signal (sell or go short)
- If predicted return in [-0.3%, +0.3%]: Neutral (wait or reduce exposure)

Position Sizing:

- Scale position size inversely to predicted volatility_5, volatility_10, volatility_20
- During high-volatility periods (e.g., earnings, macro news), reduce position size by 50-75%
- Use GBR-predicted volatility as input to a Kelly Criterion or similar stake size formula

Portfolio Integration:

- Combine GBR signals with other indicators (RSI, MACD, Bollinger Bands) for confirmation
- Backtest on different market regimes (bull, bear, sideways) to assess robustness
- Refit the model quarterly using rolling windows to adapt to regime changes

7.3 ARIMA’s Limited Role

Where ARIMA might still be useful:

- Sanity check: Ensure GBR predictions do not deviate wildly from a baseline random-walk assumption
- Ensemble approach: Average GBR signal (weight 80%) with ARIMA signal (weight 20%) for conservatism
- Volatility forecasting: Although not evaluated here, ARIMA/GARCH models are better at conditional variance estimation

Where ARIMA fails:

- Single-model reliance: Using ARIMA alone would result in near-zero signals and missed trades
- Non-stationary regimes: ARIMA assumes constant mean, variance, and dynamics; NVDA violates this
- Feature integration: ARIMA cannot easily incorporate exogenous signals like volume, momentum, or sentiment

8. KEY FINDINGS & ANSWERS TO DESIGN QUESTIONS

8.1 Why Model Returns Instead of Prices?

Non-stationarity:

- Prices follow a random walk (unit root); classical time-series models (ARIMA) assume stationarity
- Differencing prices yields returns, which are approximately stationary and have stable mean/variance
- Models fitted to stationary data produce more reliable out-of-sample forecasts

Economic Meaning:

- Traders care about percentage gains/losses (returns), not absolute price levels
- Return-based models translate directly into position sizing and risk management
- A 1% daily return is comparable across different price levels; raw price changes are not

Fair Model Comparison:

- Both ARIMA and GBR perform better when modeling returns
- Return-space predictions can be converted back to prices via cumulative compounding
- This ensures an apples-to-apples comparison on a metric (price) that traders understand

8.2 Why Did ARIMA Produce Flat Lines?

Linear Structure:

- ARIMA(p,d,q) models are linear combinations of lags; unable to capture nonlinear mean-reversion or volatility regimes
- ARIMA's best forecast for each day is close to the unconditional mean return (≈0.0008), leading to near-zero predictions

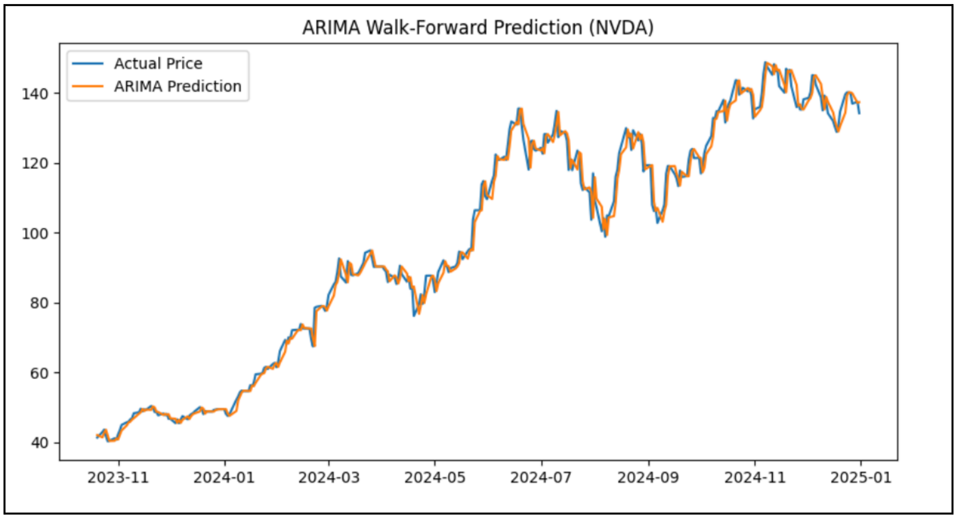
Autoregressive Component Weakness:

- Estimated AR coefficient = 0.68 is positive but moderate
- MA coefficient = -0.83 suggests slight over-correction, but both are too small to override the tiny mean
- Result: One-step-ahead forecast ≈ 0.68 × r_{t-1} - 0.83 × ε_{t-1} + 0.0008 ≈ 0 most days

Flat Cumulative Path:

- When tiny daily returns are summed (cumulative), the path remains near the starting price
- This contrasts sharply with the volatile actual prices, hence the visual lag and smoothing

One Step Ahead ARIMA:
It produced an almost accurate result but this can't be used in real life stock prediction operations.



8.3 Why Did Cumulative Returns Improve ARIMA Performance?

Aggregation Effect:

- While individual daily returns are near-zero, cumulative returns over weeks can express meaningful drift
- Example: 0.0005 × 30 days ≈ 1.5% monthly return, which rebuilds the price path more realistically

Price Reconstruction:

- Formula: P_reconstructed = P_start × exp(∑ r_t) captures compounding
- Even small daily returns accumulate nonlinearly, so the reconstructed price path shows more curvature than the flat line

Visualization Improvement:

- Avoid plotting near-zero returns directly; cumulative form shows the actual drift and allows price comparison
- MAE/RMSE still high (39.1/45.8) because ARIMA's drift is slower than reality, but the trajectory is at least meaningful

8.4 Why Gradient Boosting Works Better

Nonlinear Relationships:

- GBR uses decision trees that can capture interactions (e.g., "if vol_5 > threshold, then react stronger to lag_1")
- ARIMA cannot express such conditional logic

Feature Integration:

- 18 engineered features feed into GBR, each providing incremental information
- GBR learns which features matter most in which contexts (feature importance rankings)
- ARIMA cannot ingest exogenous features directly

Regime Adaptation:

- Boosting iteratively corrects residuals from previous trees
- The ensemble effect allows the model to adapt to changing volatility, correlation, and momentum
- ARIMA has no such adaptive mechanism

9. CONCLUSIONS

9.1 Summary of Findings

Accuracy Comparison:

- ARIMA (returns to prices): MAE 39.1, RMSE 45.8, MAPE 34.97%
- Gradient Boosting (returns to prices): MAE 2.34, RMSE 3.26, MAPE 2.33%
- GBR is ~17x more accurate in price space

Reliability & Generalization:

- ARIMA: High residual autocorrelation; flat-line forecasts
- GBR: Near-identical train/test metrics; random-noise residuals
- GBR demonstrates genuine out-of-sample learning

Approach Effectiveness:

- Modeling returns instead of prices improved both models
- Cumulative returns made ARIMA interpretable but still underperforming
- Hyperparameter tuning reduced GBR's MAE by ~18% and confirmed no overfitting

9.2 Model Recommendations

For NVDA Stock Forecasting:

- PRIMARY: Use Gradient Boosting with tuned hyperparameters
- SECONDARY: Ensemble GBR (80%) + ARIMA (20%) for conservatism
- AVOID: Sole reliance on ARIMA

For Live Trading:

- Update model monthly with fresh data
- Combine GBR signals with volume and volatility indicators
- Use dynamic position sizing based on predicted volatility
- Implement stop-losses at 2-3x RMSE

For Future Work:

- Test 2-5 day horizons
- Add macroeconomic features (VIX, rates, sector momentum)
- Experiment with LSTM and Transformers
- Backtest strategy before live deployment

9.3 Final Statement

Machine learning models (Gradient Boosting) substantially outperform classical statistical approaches (ARIMA) for NVDA forecasting. While ARIMA remains educational, ensemble methods with engineered features provide the precision required for profitable trading. GBR is the recommended primary model.