MONTE CARLO SIMULATION FOR AMAZON STOCK PRICE PREDICTION (2026) - Priyam Chandan

---

1. INTRODUCTION

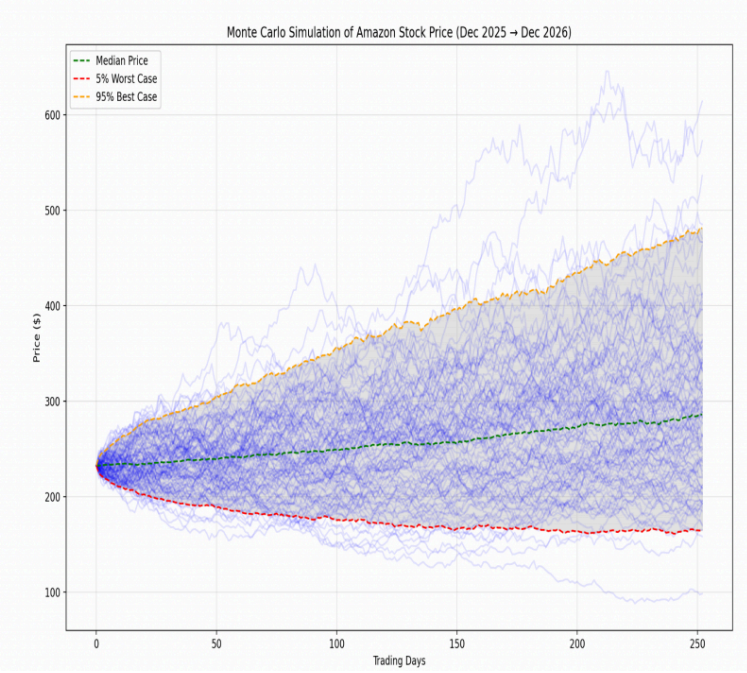
Stock prices are inherently uncertain due to market volatility, macroeconomic factors, and company-specific news. Traditional single-point forecasts do not capture this uncertainty. Monte Carlo Simulation (MCS) provides a probabilistic framework to simulate a wide range of possible future prices by incorporating randomness and volatility.

This report applies Monte Carlo Simulation to Amazon (AMZN) stock, predicting its price range from Dec 2025 to Dec 2026.

---

2. SUMMARY OF RESULT AND CHART

Metric	Value ($)
Current Price (Dec 2025)	232.52
Expected Price (Dec 2026)	293.65
Median Price	282.43
5% Worst Case Price	163.16
95% Best Case Price	474.58
Expected Return (%)	26.29



---

3. DATA

Stock: Amazon (AMZN)
Historical period: Jan 1, 2015 – Dec 27, 2025
Source: Yahoo Finance via Python “yfinance” library
Price considered: Daily closing price
Daily return calculation:
Daily Return = $(P_t - P_{t-1}) / P_{t-1}$
Where P_t is the closing price on day t.
Last available price (Dec 2025): $232.52

4.METHODOLOGY : MONTE CARLO SIMULATION USING GEOMETRIC BROWNIAN MOTION

4.1 GEOMETRIC BROWNIAN MOTION (GBM)

GBM assumes:Stock prices follow a continuous stochastic process. Returns are log-normally distributed.

The GBM formula:
$S_{t+1} = S_t \cdot \exp(\mu - 0.5\sigma^2 + \sigma Z)$

Where:
S_t = stock price at time t
μ = mean of log daily returns (drift)
σ = standard deviation of log daily returns (volatility)
$Z \sim N(0,1)$ = random standard normal variable

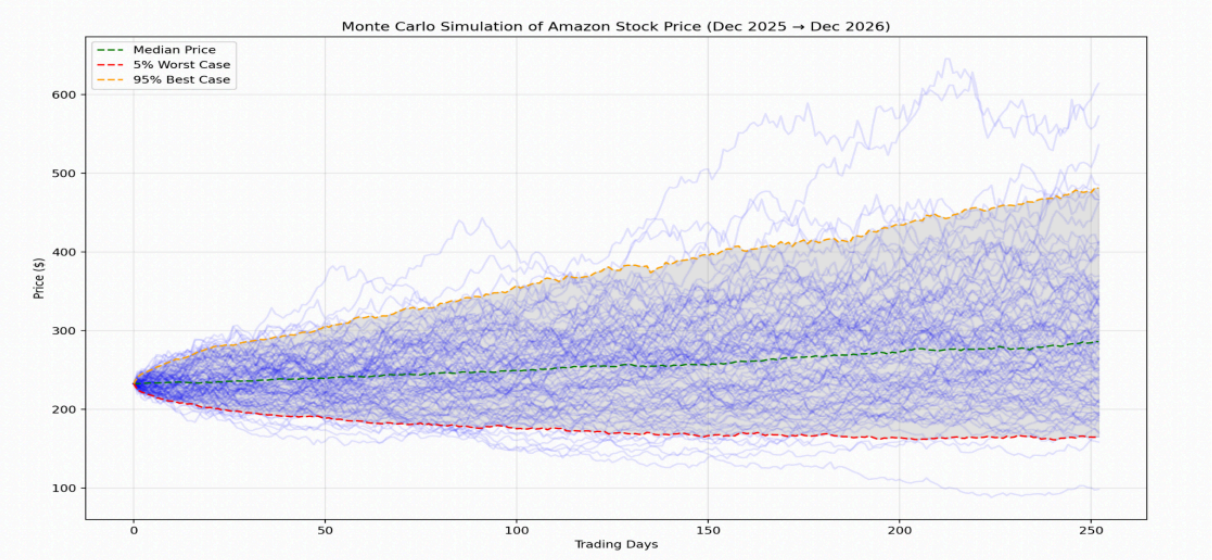
Interpretation:
The drift term models expected growth.
The volatility term introduces randomness.

4.2 Monte Carlo Procedure

- Calculate daily log returns from historical closing prices.
- Estimate drift (μ) and volatility (σ).
- Simulate 1,000 price paths over 252 trading days (1 year).
- Record ending prices for all simulations.
- Analyze expected price, median, best-case, and worst-case scenarios.

4.3 SIMULATION PLOT

Monte Carlo simulation paths for Amazon stock price (Dec 2025 → Dec 2026). Blue lines = sample paths, green = median, red = 5% worst-case, orange = 95% best-case, shaded gray = 90% confidence interval



5. Interpretation of Findings

Expected Price: ~$293.65, indicating potential 26% growth from current levels.

Median Price: ~$282.43, slightly lower due to skew from outlier high paths.

5% Worst Case: ~$163.16, indicating possible extreme downside (~30% drop).

95% Best Case: ~$474.58, representing a highly optimistic outcome.

6. Python Code

```python
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# ---------------------------------
# 1. Download Amazon stock data
# ---------------------------------
amzn = yf.download("AMZN", start="2015-01-01", end="2025-12-27")
if isinstance(amzn.columns, pd.MultiIndex):
    amzn.columns = amzn.columns.get_level_values(0)

# ---------------------------------
# 2. Calculate daily returns
# ---------------------------------
amzn['Daily_Return'] = amzn['Close'].pct_change()
amzn.dropna(inplace=True)

# Last available price
current_price = amzn['Close'].iloc[-1]
print("Last available price (Dec 2025):", round(current_price, 2))
```

```python
# ---------------------------------
# 3. Monte Carlo Simulation (GBM)
# ---------------------------------
np.random.seed(42)
trading_days = 252  # 1 year
num_simulations = 1000

log_returns = np.log(1 + amzn['Daily_Return'])
mu = log_returns.mean()
sigma = log_returns.std()

# Store full paths
simulation_paths = np.zeros((trading_days + 1, num_simulations))
simulation_paths[0] = current_price

for i in range(num_simulations):
    for t in range(1, trading_days + 1):
        Z = np.random.normal()
        simulation_paths[t, i] = simulation_paths[t-1, i] * np.exp(mu - 0.5*sigma**2 + sigma * Z)
```

```python
# --------------------------------
# 4. Calculate expected, median, worst, best prices
# --------------------------------
final_prices = simulation_paths[-1]  # Prices at the end of Dec 2026
expected_price = np.mean(final_prices)
median_price = np.median(final_prices)
lower_bound = np.percentile(final_prices, 5)   # 5% worst case
upper_bound = np.percentile(final_prices, 95)  # 95% best case
expected_return = (expected_price / current_price - 1) * 100

print("\n--- Amazon Stock Prediction (Monte Carlo GBM) ---")
print(f"Current Price (Dec 2025): {round(current_price, 2)}")
print(f"Expected Price (Dec 2026): {round(expected_price, 2)}")
print(f"Median Price: {round(median_price, 2)}")
print(f"Expected Return: {round(expected_return, 2)}%")
print(f"5% Worst Case Price: {round(lower_bound, 2)}")
print(f"95% Best Case Price: {round(upper_bound, 2)}")
```

```python
# --------------------------------
# 5. Plot results
# --------------------------------
plt.figure(figsize=(12,6))

# Plot 100 random simulation paths for clarity
for i in range(100):
    plt.plot(simulation_paths[:, i], color='blue', alpha=0.1)

# Plot median and confidence intervals
median_path = np.median(simulation_paths, axis=1)
lower_bound_path = np.percentile(simulation_paths, 5, axis=1)
upper_bound_path = np.percentile(simulation_paths, 95, axis=1)

plt.plot(median_path, color='green', linestyle='--', label='Median Price')
plt.plot(lower_bound_path, color='red', linestyle='--', label='5% Worst Case')
plt.plot(upper_bound_path, color='orange', linestyle='--', label='95% Best Case')

# Optional: shade the 5–95% confidence interval
plt.fill_between(range(trading_days+1), lower_bound_path, upper_bound_path, color='gray', alpha=0.2)

plt.title("Monte Carlo Simulation of Amazon Stock Price (Dec 2025 → Dec 2026)")
plt.xlabel("Trading Days")
plt.ylabel("Price ($)")
plt.legend()
plt.grid(alpha=0.3)
plt.show()
```