Enhancing Mean-Variance Portfolio Optimization with Stock Price Prediction Using XGBoost

December 21, 2023

Abstract

This paper explores whether accurate stock price forecasting via extreme gradient boosting (XGBoost) helps investors build an optimal financial portfolio that maximizes investment returns for their given risk tolerance. Portfolio optimiztion is a crucial task in finance as spreading investmets across a variety of assets helps reduce the risk to the investors' capital by mitigate risks associated with any single asset's underperformance. While traditional mean-variance portfolio optimization balances return and risk, its dependency on historical prices for estimating asset return distributions is a notable drawback, especially given the stock market's volatility. This paper introduces a novel method for portfolio optimization using future price estimates predicted by XGBoost. The effectiveness of this approach is assessed through backtesting analysis, comparing the performance of portfolios optimized with XGBoost-predicted prices against those built using historical data. The results reveal that XGBoost-based portfolios yield higher cumulative returns, particularly over longer future periods, demonstrating the benefits of incorporating predictive modeling into portfolio optimization for enhanced returns and risk management.

Keywords: Time Series Analysis, Gradient Boosting, Portfolio Management

1 Introduction

Portfolio management strategically allocates capital among various assets to maximize investor utility, typically gauged by return and risk tolerance. The earliest study in the portfolio optimization field dates back to the 1950's with the mean-variance model proposed by Harry Markowitz, which seeks the optimal trade-off between the return and risk measured by the mean and the variance of the portfolio returns [Chen et al., 2021]. Despite its widespread application, the model's primary limitation is its reliance on historical prices for future return and volatility estimates, a concern in the notably volatile stock market. Enhancing portfolio optimization with stock price predictions could lead to more robust future market performance. While predicting stock prices remains a challenging task, recent advancements in machine learning, including neural networks and ensemble models, have shown promising results in addressing this challenge.

Ensemble learning models are developed to reduce bias and variance and achieve better stability than a single model like decision trees. In particular, boosting focuses on prediction improvement by strategically combining the output of weak learners to create a strong learner. One of the popular boosting models is gradient boosting that fits new models to the residual errors made by the previous models. This approach ensures each new model to be more effective than the previous one by correcting its errors. However, gradient boosting technique is inherently slow as it sequentially adds a model to an ensemble [Dezhkam and Manzuri, 2023]. There are various implementations of gradient boosting algorithms that enhance their training speed, such as Gradient Boosting Machine (GBM) and Extreme Gradient Boosting (XGBoost). Among them, XGBoost has gained reputation for its exceptional performance and computation speed achieved through regularization and distributed computing in various Kaggle competitions and literature in a wide variety of areas such as health care, finance and environmental science, introduced in Section 2 [Ogunleye and Wang, 2019].

This paper proposes a novel portfolio optimization approach that predicts future stock prices using XGBoost to form the optimal portfolio. Finding the optimal composition of financial portfolios is a crucial downstream task of stock price forecasting as success of portfolios in the volatile stock market greatly hinges upon accurate estimation of stock returns when constructing them. While modern portfolio theory (further discussed in 3.3) is a well-regarded approach for balancing return and risk in portfolio optimization, it estimates the probability distribution of returns based on historical returns. This may not be a good estimate when the future stock market behaves differently. Hence, it is worthwhile to explore whether time series forecasting enhances the traditional approach. Does the iterative approach of gradient boosting work for time series data? Does accurately predicting the stock market behavior directly translate into the increased investment returns?

This paper answers these questions by applying XGBoost to Standard & Poor 500 stock market data to predict the next day's price and determining the optimal portfolio composition using these predictions, and backtesting the cumulative returns of the portfolio to examine how it would have behaved in the past stock market. The proposed approach is further discussed in the following sections. Section 2 introduces applications of XGBoost in literature. Section 3 introduces the key statistical techniques underpinning this project, including extreme gradient boosting and time series analysis, complemented by an overview of modern portfolio theory to establish a foundation for the financial application. Section

4 details the data, model development, and evaluation in comparison to the benchmark model, random forest. Finally, Section 5 concludes the paper, offering insights into potential future work in this area.

2 Related Work

This section highlights the diverse applications of XGBoost found in literature, showcasing its versatility in handling both classification and regression tasks across various fields. A notable example is the work of Ogunleye and Wang [2019], who utilized XGBoost for diagnosing Chronic Kidney Disease (CKD), a condition affecting over 10% of the global population. The early detection of CKD is critical in preventing its advancement to end-stage, which could reduce dependence on extensive treatments like dialysis. Remarkably, XGBoost has demonstrated a 100% accuracy in CKD diagnosis, highlighting its potential to improve healthcare outcomes by significantly reducing both the costs and time associated with diagnosing CKD patients.

The versatility of XGBoost extends into finance, where it has demonstrated outstanding performance in forecasting stock market trends and Bitcoin price movements [Nazareth and Reddy, 2023, Uras and Ortu [2021]]. Additionally, [Yun et al., 2021] successfully applied XGBoost for predicting stock price direction changes, achieving a notable 93.82% accuracy on the Korean stock market index data. Enhanced feature engineering techniques were used in tandem to improve the performance through genetic algorithm and additional technical indicators, which will also be used in this paper.

Despite its growing popularity, the application of XGBoost in portfolio optimization is still relatively underexplored. [Chen et al., 2021] predicts stock prices for portfolio optimization by using XGBoost with an improved version of the firefly optimization algorithm (IFA) for hyperparameter tuning. Their approach that combined XGBoost, IFA, and mean-variance (MV) portfolio optimization (explained in Section 3.3), led to a remarkable increase in cumulative return, achieving 94% compared to just 31% with the standard XGBoost + MV model. [Dezhkam and Manzuri, 2023] demonstrates the effectiveness of XGBoost in predicting stock price change directions (i.e. whether it will rise or fall), which is a classification problem. It uses the results to construct portfolios with stocks whose price is expected to rise. They enhanced the model's performance by incorporating the Hilbert-Huang Transform, a method for handling non-linear and non-stationary data, to better analyze closing price information. This adaptation significantly boosted the classification accuracy.

Existing literature on portfolio optimization using XGBoost typically focuses on comparing the performance of XGBoost-based portfolios with other forecasting models and equal weight portfolios, as seen in studies by Wang [2022] and Jidong and Ran [2018]. However, this paper aims to broaden these comparisons to highlight the importance of using future prices in portfolio optimization. It not only assesses XGBoost-derived portfolios against another statistical model, such as random forest, but also contrasts them with traditional portfolio optimization method that relies on historical price data. This dual comparison is designed to underscore the potential of XGBoost as a time series forecasting model in enhancing the classic mean-variance optimization technique.

3 Methods

This section provides an exposition of the methods used in this paper. Section 3.1 discusses the unique characteristics of time series data. We also briefly discuss technical indicators that are used to enhance the model's prediction accuracy by capturing the essential temporal features of the stock data. Then, Section 3.2 explains how XGBoost performs predictions through its enhanced implementation of gradient boosting. Section 3.3 introduces modern portfolio theory, illustrating how to determine optimal portfolio composition by calculating return and risk from future prices.

3.1 Time Series Analysis

Time series data contains values recorded at different time steps, usually at regular intervals. Time series forecasting techniques address questions about future values, such as predicting outcomes for the next day or the upcoming 14 days. Univariate forecasting methods, like ARIMA, tackle these questions by analyzing the inherent seasonality and trend within the time series. However, real-world scenarios are typically more intricate, influenced by a variety of factors. Multivariate time series data, which includes multiple variables at each time step, requires more complex forecasting methods. These multivariate techniques address questions involving interactions between various factors over time. In the following sections, we delve into fundamental temporal patterns and explore technical indicators for capturing intricate temporal dynamics.

3.1.1 Seasonality and Autocorrelation

Seasonality in time series data refers to recurring patterns at regular time intervals. Detecting strong seasonality is advantageous for predictive modeling because it suggests that forecasting future values can be reasonably accurate by copying values from the previous season. When a time series is correlated with a lagged version of itself, it is considered autocorrelated.

The Partial Autocorrelation Function (PACF) measures the extent of the autocorrelation of a variable at different points in time while controlling for relationships at shorter intervals. In a PACF plot, bars that significantly deviate from a threshold line (usually marked for statistical significance) indicate a strong relationship at that specific lag, after accounting for relationships at shorter lags. Stock prices often show significant PACF only up to lag 1 or 2 (See Figure 1). This indicates that the price from at most two days ago helps predict the current day's price, but no further. We thus include lagged price features up to 2 days as predictor variables in our model. We also added date-extracted features such as day of the week, month, quarter, or year to help detect seasonal patterns in the data.

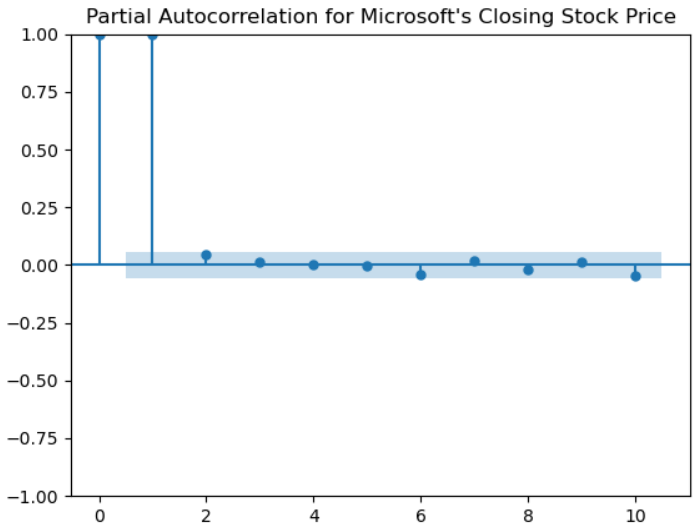

Figure 1: Partial Autocorrelation Plot (stock price data was retrieved using yfinance package and PACF plot was created using statsmodels package) [Ran, 2023, Seabold and Perktold [2010]]

3.1.2 Tehcnical Indicators

This complex temporal nature evidently calls for more than just a surface-level examination of the raw data when analyzing stock prices. This is where technical indicators come into play. These tools process and refine raw price data to underscore underlying information that might not be readily apparent. For example, moving averages help focus on longer-term trends by smoothing out short-term fluctuations. It achieves this by calculating an average value for a subset of data points within a specified time window or period as it "moves" through the time series. Technical indicators help provide diverse perspectives on the market, thereby enhancing the accuracy of predictive models. In [Yun et al., 2021], the most significant accuracy increment in predicting the stock price change direction came from adding 67 technical indicators to the original historical stock price data. Table 1 lists 11 technical indicators used in this project that are widely used both in academia and the finance industry [Jiang et al., 2020]. Using different parameter settings, we create a total of 17 versions of different indicators, which are adopted as predictors for the XGBoost model. The full description of each indicator is provided in the Appendix A.

3.2 Extreme Gradient Boosting (XGBoost)

XGBoost is one of the fastest and most efficient implementations of gradient boosting algorithms with a few modifications in the objective function. Below is a brief exposition of the XGBoost model proposed by the seminal paper [Chen and Guestrin, 2016].

Assume there is a data set D with n samples and m variables.

$$D = (x_i, y_i), x_i \in \mathbb{R}^m, y_i \in \mathbb{R},$$

Table 1: Technical Indicators and Parameter Setting.

Technical_Indicators	Parameter
Simple Moving Average	2, 7, 9, 14, 30
Exponential Moving Average	9
TrueRange	N/A
Average True Range	14
Percentage Rate of Change	9
Stochastic Oscilator	14
Williams Percentage Range	14
Relative Strength Index	14
Moving Average Convergence/Divergence (MACD)	12,26
MACD Signal	9
Bollinger Bands	20
On Balance Volume	N/A

where x_i is a vector of m predictors and y_i is one-dimensional target vector associated with x_i.

A tree ensemble model uses K additive functions f_k to predict the output:

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i),\ f_k \in \mathcal{F} \tag{1}$$

where ϕ is defined as the sum of functions f_k and $\mathcal{F}$ is the set of regression trees.

$$\mathcal{F} = \{f(x) = w_{q(x)}\},\ q : \mathbb{R}^m \to T, w \in \mathbb{R}^T, \tag{2}$$

where q is the structure of each tree, T is the number of leaves in the tree, and w is leaf weights. Each f_k is a function that maps a data point to a continuous score of the leaf that it reaches in each tree structure q. Typically, this score represents the average outcome of all data points within that leaf. The score of this leaf then serves as the prediction for the target variable of the data point that ended up in the leaf. The ensemble model, as described in Eq. 1, computes the final prediction for a given data point x_i by summing the corresponding leaf weights w_i in K trees, each represented by a function. Figure 2 illustrates how the final prediction for a given observation can be obtained.

XGBoost learns the set of functions f_k in Eq. 1 that constitute the model by minimizing the following regularized objective proposed in [Chen and Guestrin, 2016]:

$$L(\phi) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k). \tag{3}$$

Ω is a regularization term that can be calculated as follows:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda ||w||^2,$$

where γ is an L1 regularization coefficient, λ is an L2 regularization coefficient, T is the number of leaves, and w is leaf weights. l is a differentiable convex loss function that measures the difference between the prediction $\hat{y}_i$ and the target y_i. The first term in Ω

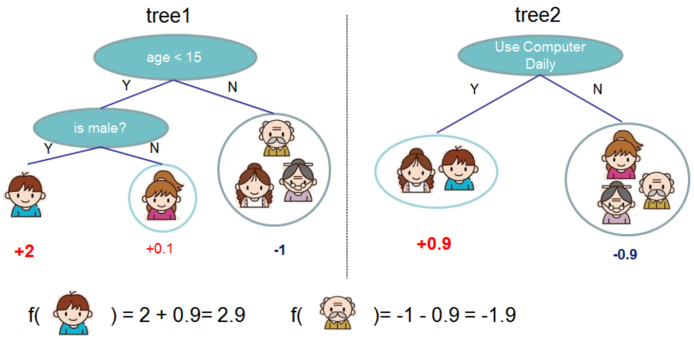

Figure 1: Tree Ensemble Model. The final prediction for a given example is the sum of predictions from each tree.

Figure 2: Final Prediction in Regression Tree Ensemble (Figure 1 from the original paper [Chen and Guestrin, 2016])

penalizes the complexity of the model, and the second term helps smooth the final learnt weights to avoid overfitting. Hence, this regularization helps select a model that uses simple and predictive functions.

Recall that the model is trained in an additive manner as illustrated in Eq. 1. Let $\hat{y}_i{}^{(t)}$ be the prediction of the i-th instance at the t-th iteration. The algorithm greedily adds f_t at t-th step that minimize the objective (Eq. 3), which can be rewritten as follows:

$$L^{(t)} = \sum_i l(y_i, \hat{y}_i{}^{(t-1)}) + f_t(x_i)) + \Omega(f_t). \tag{4}$$

Chen and Guestrin [2016] state that the second-order Taylor expansion of Eq. 4 is performed to quickly optimize the objective:

$$L^{(t)} \approx \sum_i l(y_i, \hat{y}_i{}^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)) + \Omega(f_t), \tag{5}$$

where $g_i = \partial_{\hat{y}_i{}^{(t-1)}} l(y_i, \hat{y}_i{}^{(t-1)})$ and $h_i = \partial^2_{\hat{y}_i{}^{(t-1)}} l(y_i, \hat{y}_i{}^{(t-1)})$ are the first and second derivatives on the loss function, respectively. The name "gradient boosting" comes from the use of gradient descent optimization in the process. The idea is to find the direction in which modifying the predictions will most reduce the overall prediction error. New trees are added to move the model's predictions in the direction of descending gradient.

3.3 Modern Portfolio Theory

Portfolios are a collection of investment instruments, including stocks, bonds, commodities and cash. They play a crucial role in hedging risks for investors since spreading investments across a variety of assets may mitigate the impact of poor performance in any single investment. When the value of one asset falls, this decline can be offset by gains in other assets, if their prices move in opposite directions.

The fundamental goal of constructing a portfolio is to maximize expected returns within the bounds of an investor's risk tolerance. Modern Portfolio Theory (MPT), or mean-

variance model, posits that combining various risky assets can yield a portfolio whose overall expected return mirrors that of its individual components, but with significantly reduced risk [Fabozzi et al., 2008].

MPT offers a mathematical framework to assemble these risky assets in a way that the expected return is maximized for a given level of risk. A risky asset is one for which the return that will be realized in the future is uncertain. For instance, an investor buying Microsoft stock today without intending to sell it immediately cannot predict the exact return at the end of the holding period. This return will depend on the future stock market behavior. Given this unpredictable nature, MPT models the returns of risky assets as random variables [Markowitz, 1952].

Assume an investment instrument i was purchased at price x_0 and sold at price x_1. The rate of return on this asset is

$$R_i = \frac{x_1 - x_0}{x_0} = \frac{x_1}{x_0} - 1. \tag{6}$$

It is the fractional change between the current and a prior price. For example, a daily return of 0.3 means the price rose 30% compared to the previous day. The terms "rate of return" and "return" are often used interchangeably in the literature.

Let us now construct a portfolio with n assets. Let w_i be the proportion of asset i in the portfolio. Markowitz expresses the return on the portfolio as a weighted sum of the returns of the individual assets [Markowitz, 1952]:

$$R_p = \sum_{i=1}^{n} w_i R_i, \quad s.t. \sum_{i=1}^{n} w_i = 1. \tag{7}$$

Let μ_i be the expected value of R_i. The expected return on the portfolio is simply the weighted sum of the expected values of the individual returns:

$$E[R_p] = \sum_{i=1}^{n} w_i E[R_i] = \sum_{i=1}^{n} w_i \mu_i.$$

To calculate the expected value $E[R_i]$, the probability distribution function (PDF) of the rate of return must be defined. MPT estimates the PDF based on historical returns of a risky asset in the past [Fabozzi et al., 2008]. The limitation of this estimation will be further discussed at the end of this section, calling for stock price forecasting in this project.

Let σ_{ij} be the covariance between R_i and R_j, two different investment instrument returns, and σ_i^2, the variance of R_i. Markowitz argued that variance (or standard deviation) measures the riskiness of an investment. The risk of the portfolio as a whole is defined as the variance of returns R_p, equivalent to the sum of variances of the returns of the individual assets and covariances among the investments:

$$V(R_p) = \sum_{i=1}^{n} w_i^2 \sigma_i^2 + 2 \sum_{i=1}^{n} \sum_{j>1}^{n} w_i w_j \sigma_{ij}. \tag{8}$$

Eq. 8 can be expressed in shorthand notation as

$$V(R_p) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij}.$$

by combining the indices.

The correlation coefficient between the return of two assets i and j is then :

$$\rho = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$$

.

The key contribution of MPT lies in its emphasis on the correlation between asset returns when forming a portfolio to achieve effective diversification [Markowitz, 1952]. For instance, companies within the same industry often experience simultaneous downturns more frequently than those in varied industries. Markowitz thus advocated for the combination of assets with lower positive correlation, which can reduce the overall risk of the portfolio without necessarily compromising on the expected returns.

It is important, however, to understand that achieving the highest possible return with the lowest possible risk is not typically feasible. There is an inherent trade-off between return and risk: a portfolio with lower risk is usually accompanied by lower returns, while a higher risk can lead to higher returns. The essence of portfolio optimization, therefore, lies in maximizing the expected return for an acceptable level of risk as determined by the investor's risk tolerance. Portfolios that achieve this balance are known as efficient portfolios and collectively form what is called the efficient frontier. This concept is visually represented in Figure 3, where the efficient frontier's positive slope (the segment connecting points II and III) illustrates the return-risk trade-off. Each portfolio on this frontier is deemed 'optimal' for its specific level of risk. Investors typically select a portfolio from this frontier that aligns with their risk tolerance. Portfolios falling below this frontier line are considered inefficient or sub-optimal, as they yield lower returns for their level of risk than is theoretically possible. Any portfolio positioned above this line is regarded as unattainable [Fabozzi et al., 2008].

In order to balance both return and risk in our portfolio formation, we set Sharpe ratio as our optimization objective:

$$\frac{(E[R_p] - R_f)}{\sigma_p},$$

where R_f is the return on a risk-free asset such as U.S. government bond. It is constant because one can precisely anticipate the amount of the realized returns on these assets. 3-month Treasury bill (T-bill) rate is often used as a benchmark for the risk-free rate [Hayes]. T-bills, short-term government debt securities with maturities of one year or less, are regarded as low-risk investments due to the credibility and backing of the U.S. government. Sharpe ratio measures the additional return an investor receives for taking on the risk associated with their investment compared to a risk-free alternative. This metric tells us whether the additional return achieved by taking on additional risk is justified and helps in comparing the returns on different portfolios on a risk-adjusted basis.

MPT still stands as a pivotal framework in portfolio optimization, enabling investors to simultaneously consider both return and risk in their investment decisions. However,

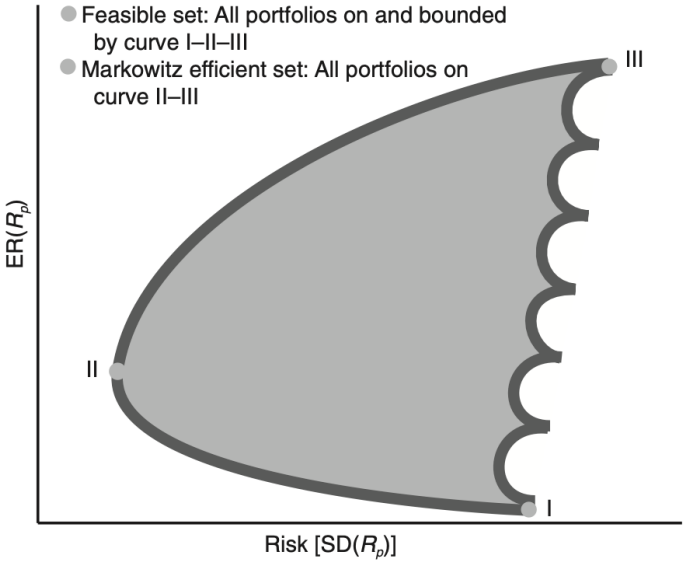

Figure 1.3 Feasible and Efficient Portfolios with More Than Two Assets*

Figure 3: Efficient Frontier [Fabozzi et al., 2008]

a notable limitation of MPT is its reliance on historical data to estimate the probability distribution of returns. Such an approach might not always be reliable, as past market trends don't necessarily predict future movements accurately. To overcome this challenge, this paper proposes the use of the XGBoost model in predicting stock prices used for portfolio optimization. As seen in Section 2, it is adept at predicting future stock prices, thereby providing a more accurate estimation of future expected returns and risks.

4 Applications

In this section, we build XGBoost models to predict stock prices of nine companies from the Standard & Poor's 500 index, analyzing data spanning the past ten years. Our process begins with refining raw stock price data to develop technical indicators, capturing key temporal market patterns. Through cross-validation, we tune the hyperparameters of our XGBoost model. Once the optimal parameters are determined, we train the model and forecast next-day stock prices for the out-of-sample period. These forecasts are then benchmarked against predictions from a random forest model. Utilizing the XGBoost-generated price predictions, we calculate the Sharpe ratio for portfolio optimization. The derived optimal weights are subsequently tested against actual market returns in an out-of-sample period through backtesting, which assesses the strategy's effectiveness by simulating its performance with historical data. We will compare our portfolio's performance with various benchmarks: an equal weights portfolio, numerous portfolios generated through 10,000 Monte Carlo simulations, and a traditional MPT approach based on historical prices. The entire methodology is visually summarized in Figure 4.

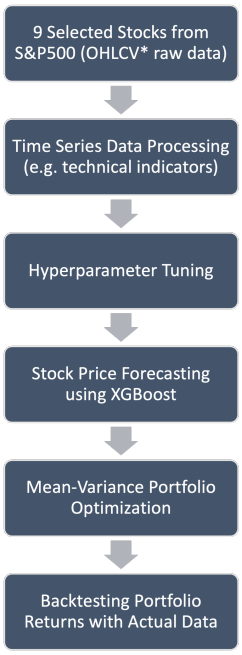

Figure 4: Proposed Method.

4.1 Data

The application uses stock price data of 9 stocks selected from Standard & Poor 500 (S&P 500) across various sectors (See Table 3). S&P 500 is a stock market index that measures the performance of 500 large companies listed on stock exchanges in the United States, serving as a benchmark for the U.S. stock market. The data was retrieved using **yfinance** package in Python from 12/19/2013 to 12/19/2023 The raw data includes Open/High/Low/Close price and Volume data for each stock (See Table 2). The target variable for the forecasting task is close price, and the raw variables are pre-processed to generate 26 predictor variables, including the 17 technical indicators, 2 lagged target variables, and 7 date-extracted features explained in Section 3.1. Note that the target variable is moved up by a single row to align the predictors with the next day's price, avoiding look-ahead bias during prediction. For each stock, the first 30 days and the last days were dropped because moving average operations for creating technical indicators and shifting the target value by 1 row created missing values. The final data set contains 2486 observations and 27 variables for each stock.

Table 2: Variables in the Raw Data.

Variable	Meaning
Open Price	Price at which a stock first trades when the markets open for the trading day.
High Price	Highest price at which the stock trades during the trading day.
Low Price	Lowest price at which the stock trades during the trading day.
Close Price	Final price at which the stock trades at the end of the trading day.
Volume	Total number of shares of the stock that are traded during the trading day.

Table 3: Assets selected for portfolio formation. They will be referred by their ticker symbols.

Ticker	Stock	Sector
MSFT	Microsoft Corporation	Technology
AMZN	Amazon.com, Inc.	Consumer Cyclical
UNH	UnitedHealth Group Incorporated	Healthcare
AAPL	Apple Inc.	Technology
NVDA	NVIDIA Corporation	Technology
BRK-B	Berkshire Hathaway Inc.	Financial Services
JPM	JPMorgan Chase & Co.	Financial Services
JNJ	Johnson & Johnson	Healthcare
XOM	Exxon Mobil Corporation	Energy

The returns of all 9 stocks are roughly normally distributed, with a slightly high kurtosis. Only two of them are displayed in Figure 5. Table 4 shows the expected return and risk measured by the average and standard deviation of returns, respectively. JNJ has the lowest return and risk, whereas NVDA has the highest return and risk. This relationship clearly shows the inevitable trade-off between return and risk discussed in Section 3.3.

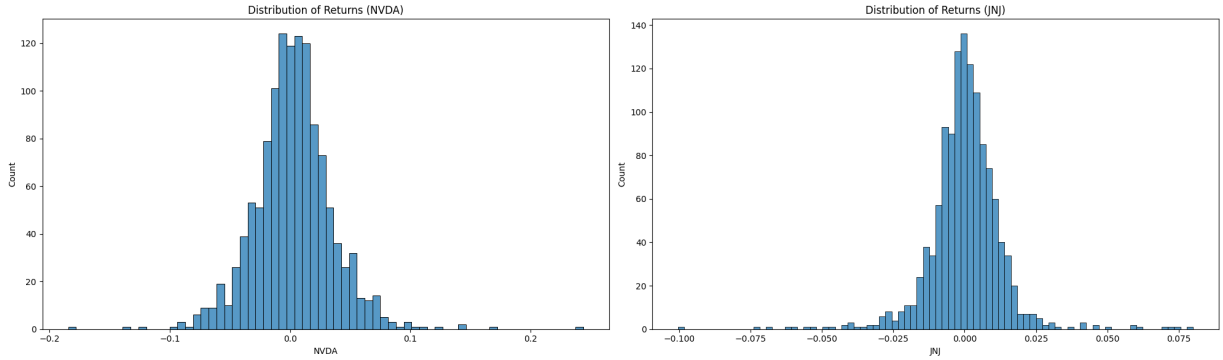

Figure 5: Distribution of Rate of Return

Table 4: Expected return and risk of individual assets. The assets are sorted by the expected return in ascending order

Ticker	Expected.Return	Risk
JNJ	0.0001254	0.0129622
XOM	0.0004404	0.0217738
BRK-B	0.0005534	0.0138293
JPM	0.0005713	0.0201861
AMZN	0.0007067	0.0225028
UNH	0.0007331	0.0186826
MSFT	0.0011661	0.0194069
AAPL	0.0014230	0.0204948
NVDA	0.0025566	0.0327044

Indeed, NVDA's closing price has been very volatile in the past 5 years, whereas JNJ has been steady around $150 throughout the time (See Figure 6). NVDA' has shown's returns are wildly fluctuating, worst in the beginning of 2020, presumably when the stock market was turbulent in the beginning of the pandemic. JNJ's daily returns were also affected during the same period, but not as severely as NVDA (See Figure 7).

Figure 6: Clsoing Price

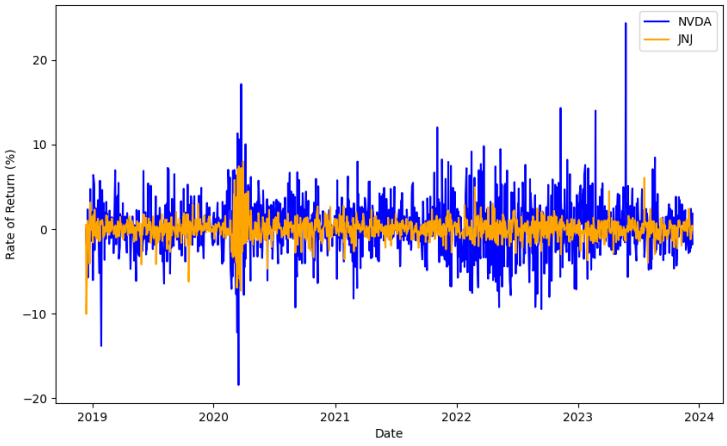

Figure 7: Daily Returns

4.2 Stock Price Forecasting Using XGBoost

4.2.1 Hyperparameter Tuning and Model Training

The XGBoost model was built using the **xgboost** package developed by [Chen and Guestrin, 2016]. Hyperparameter tuning and model training were done independently for each of 9 stocks. Each model was trained on the data from 2/5/2014 to 12/30/2022; testing period

spans from 1/3/2023 to 12/19/2023. Figure 8 illustrates the timeline of training, testing, and backtesting.

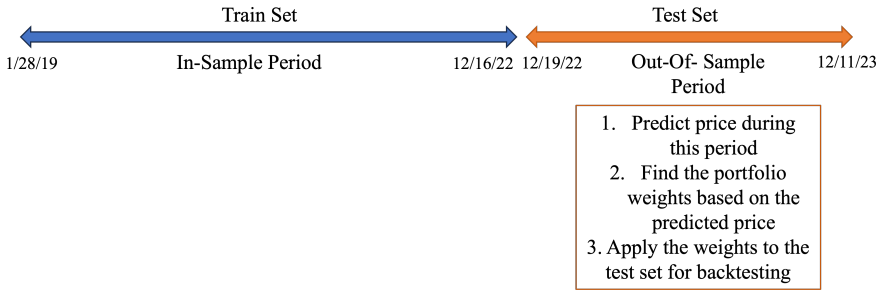

Figure 8: Timeline of Model Training, Testing, and Backtesting Phases

XGBoost requires many hyperparameters and prediction accuracy is sensitive to the hyperparameter setting. We employed the Optuna optimizer, an advanced hyperparameter tuning tool, to determine the best hyperparameters from 100 trials, each involving 5-fold cross-validation (CV). It operates by suggesting sets of hyperparameters, testing them through an objective function (in this case, root mean squared error of the predictions under 5-fold CV), and then iteratively refining these suggestions based on previous results. It intelligently navigates the hyperparameter space using Bayesian optimization to efficiently identify the most promising configurations.

Note that the traditional k-fold CV assumes the data is independent and identically distributed. However, this assumption is violated in time series data and it is critical to maintain the temporal order to prevent look-ahead bias during the validation process. Thus, we used a time series cross-validator `TimeSeriesSplit` in scikit-learn that ensures each split contains a contiguous block of samples for training and testing while maintaining the temporal order of the data (See Figure 9).

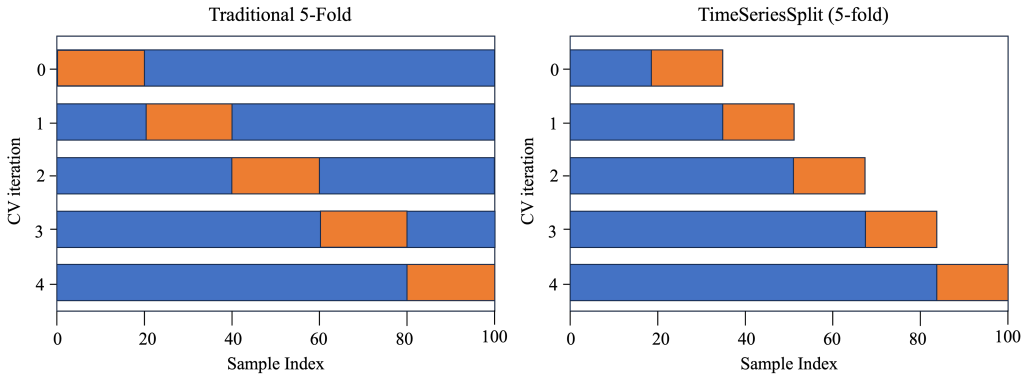

Figure 9: Difference between traditional k-fold cross validation and Scikit-Learn's TimeSeriesSplit approach. I adapted the plot from a visualization provided by Scikit-learn [scikit learn, 2023c]. The blue bar represents train set and the orange bar the validate set.

Table 5: List of tuned hyperparameters for XGBoost

Variable	Meaning
Open Price	Price at which a stock first trades when the markets open for the trading day.
High Price	Highest price at which the stock trades during the course of the trading day.
Low Price	Lowest price at which the stock trades during the trading day.
Close Price	Final price at which the stock trades at the end of the trading day.
Volume	Total number of shares of the stock that are traded during the trading day.

4.2.2 Model Evaluation

The model's performance on the test set is evaluated based on two measures of prediction accuracy, root mean squared error (RMSE) and mean absolute percentage error (MAPE). RMSE,

$$\sqrt{\frac{1}{n}(\sum_i (y^i_{true} - y^i_{pred})^2)},$$

gives greater weight to larger errors by squaring them, but the scale is converted back to the same units as the original data, making it more interpretable than MSE. This metric was chosen since larger errors would have serious consequences in the real-world financial forecasting. On the other hand, MAPE expresses average magnitude of errors as a percentage relative to the actual values,

$$\frac{1}{n}(\sum_i |\frac{y^i_{true} - y^i_{pred}}{y^i_{true}}|).$$

It thus provides more context and is useful for comparing the performance of models across different scales.

XGBoost's performance will be compared to random forests, the baseline model. Both random forest and XGBoost are ensemble techniques designed to overcome the limitations of individual decision tree models. Random forest builds robust models by developing large trees from bootstrapped samples and resolves the greediness of decision trees by considering only a subset of predictors at each split. While boosting primarily aims at reducing bias, random forest focuses on variance reduction. Given that this project is centered around forecasting, it's particularly interesting to assess if the boosting model indeed yields more precise predictions compared to random forest. Note that we have bypassed hyperparameter tuning for the random forest model in this study. However, random forests are generally less sensitive to hyperparameters than boosting models.

The performance of the two models for the back-test period is shown in Table 6. Surprisingly, random forest outperforms XGBoost on half of the stocks. Both models did worst in NVDA with predictions being off by approximately 20% on average. Both models achieved their best performance with JNJ, which was the least risky asset in our portfolio, showing discrepancies of less than $2 on average. Although the performance gap between the two models is marginal, XGBoost's underperformance might stem from its sensitivity to hyperparameter settings, as suggested by [Dezhkam and Manzuri, 2023].

Table 6: Prediction Accuracy Comparison between XGBoost and Random Forest

Ticker	MAPE_xgboost	MAPE_rf	RMSE_xgboost	RMSE_rf
NVDA	0.2190274	0.2086378	120.994602	116.209769
JPM	0.0124927	0.0127338	2.511331	2.453727
JNJ	0.0081085	0.0082046	1.763887	1.798372
MSFT	0.0430407	0.0383267	19.583367	18.327324
XOM	0.0658461	0.0445978	8.341909	6.274079
AMZN	0.0180242	0.0185628	2.781127	2.852300
UNH	0.0130037	0.0132748	8.519568	8.768049
BRK-B	0.0198994	0.0247311	9.340410	11.572177
AAPL	0.0413294	0.0381704	10.616325	9.927185

4.2.3 Mean-Variance Portfolio Optimization

In this section, we use predicted future prices to construct optimal portfolios that are more aligned with upcoming market realities. We conduct a comparative analysis with the traditional optimization approach based on historical prices. We used Sequential Least Squares Programming, an optimization algorithm, to find the best mix of assets that maximize the Sharpe ratio. It does so by iteratively adjusting asset weights to minimize a cost function, negative Sharpe ratio [Dezhkam and Manzuri, 2023]. In calculating the Sharpe ratio, the risk-free rate of 0.056 was used, the 3-month Treasury-bill rate for the back-test period retrieved from [FRE, 2023]. Tables 7 shows the final weights obtained by the optimization for both strategies.

Table 7: XGBoost-Based Portfolio Allocation

Ticker	XGBoost_Weight	MPT_Weight
NVDA	0.2803791	0.3608963
JPM	0.1168139	0.0000000
JNJ	0.0000000	0.0000000
MSFT	0.1495221	0.0374712
XOM	0.0000000	0.0000000
AMZN	0.1480239	0.0000000
UNH	0.0000000	0.5550751
BRK-B	0.0000000	0.0000000
AAPL	0.3052610	0.0465573

The portfolio created using XGBoost is presented on the efficient frontier in Figure 10. Its position on the frontier signifies that the optimization effectively identified weights that maximize risk-adjusted returns. For comparative context, we simulated 10,000 portfolios with randomly chosen weights, the majority of which are positioned below the efficient frontier. We have not included the portfolio from the traditional approach in this display, as it is based on a distinct set of return and risk calculations compared to the forecasting approach, resulting in a separate efficient frontier.

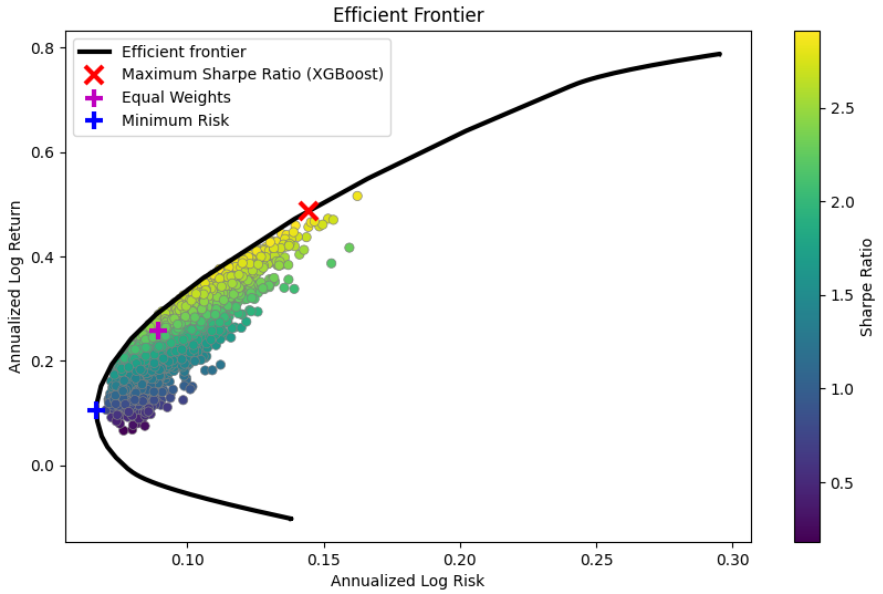

Figure 10: Efficient Frontier

4.3 Portfolio Performance

In this section, we compare the performance of three portfolios formed by three strategies: XGBoost, traditional MV model, and equal weights. The annual returns, risks, and Sharpe ratios for each strategy are shwon in Table 8. Notably, the portfolio based on XGBoost predictions exhibits the highest annual Sharpe ratio of 3.036. A Sharpe ratio below 1 is considered suboptimal, while a ratio between 1 and 1.99 is adequate to good, 2 to 2.99 is very good, and above 3 is excellent. Interestingly, the traditional approach lags behind even the equal weights strategy.

Table 8: Annualized return, risk, and sharpe ratio comparison of 3 portfolio strategies

Strategy	Annual.Return	Annual.Risk	Annual.Sharpe.Ratio
XGBoost_Sharpe	0.6279339	0.1444189	3.0100749
Equal Weights	0.2955401	0.0893883	2.3082193
MPT_Sharpe	0.3505451	0.2720521	0.9112528

Fig 11 visualizes the back-tested cumulative return of three strategies during the out-of-sample period. The performance of the XGBoost-based portfolio strategy notably surpasses others, with the advantage becoming more pronounced the further we project into the future. It shows a compound return of 100% after a year, which means the initial amount of investment is now doubled. This result suggests that a portfolio strategy grounded in future market predictions, rather than historical prices, is evidently more effective in generating better returns.

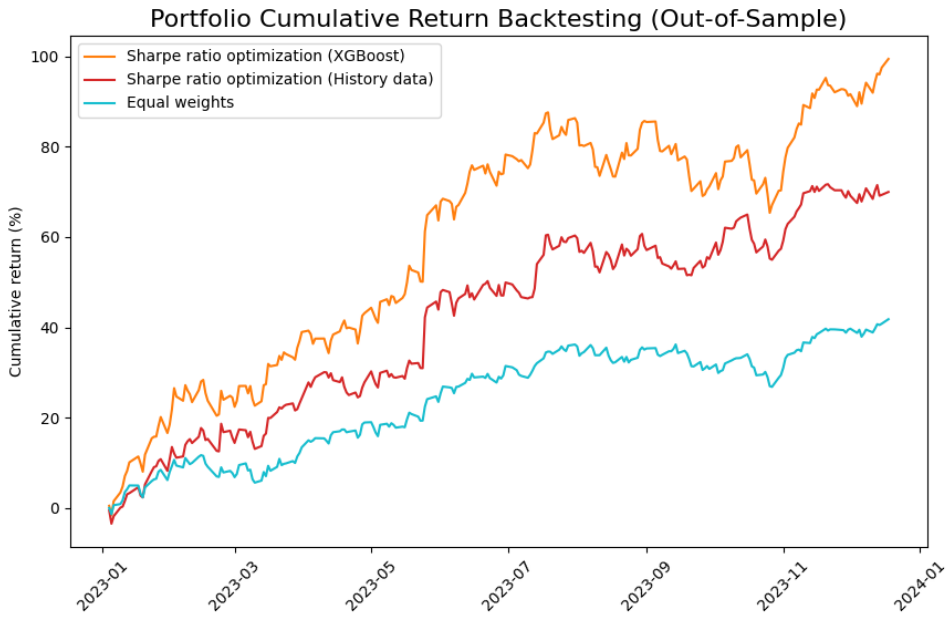

Figure 11: Out-Of-Sample Cumulative Returns

5 Conclusion

In this paper, we have delved into the applications of XGBoost in stock market prediction. Utilizing the predictive capabilities of the gradient boosting model, our approach has shown a notable edge over traditional portfolio strategies that depend on historical price data. However, our findings also bring to light some limitations. Notably, our version of the XGBoost model only showed marginal improvement in performance than the baseline random forest model. This may be attributable to boosting algorithms' sensitivity to hyperparameters. This finding suggests valuable insights into model selection. Although gradient boosting is renowned for its accuracy through bias reduction, random forest could serve as a better alternative if it achieves similar performance with no hyperparameter tuning.

Looking ahead, our future endeavors will involve exploring more dynamic and flexible approaches to portfolio optimization that better reflect real-world scenarios. The factors to consider include investment durationk, transaction costs and investment strategies, such as long/short positions. (Long positions involve buying and holding assets with the expectation that their value will increase, while short positions involve borrowing and selling assets with the hope of repurchasing them at a lower price, profiting from the difference.)

The implications of this project extend beyond its immediate results. It not only demonstrates the potential of XGBoost in accurately forecasting future stock prices but also opens up avenues for its application in various other financial tasks. Accurate forecasting is a valuable asset in finance, and portfolio optimization is just one of many areas that stand to benefit from these advancements. As we continue to refine our models and methods, the prospects for enhancing financial decision-making and strategy development through technology appear increasingly promising.

A Appendix

We provide below the brief description of the 11 employed technical indicators [Jiang et al., 2020].

1. Simple Moving Average (SMA): A technical indicator that calculates the average of a selected range of prices, usually closing prices, by the number of periods in that range.

2. True Range: Measures the volatility of an asset by considering the largest range of a trading period, incorporating the current high minus the current low, the absolute value of the current high minus the previous close, and the absolute value of the current low minus the previous close.

3. Average True Range (ATR): A moving average of the true ranges mentioned above, ATR is used to measure volatility.

4. Percentage Rate of Change: This momentum indicator measures the percentage change in price between the current price and the price a certain number of periods ago.

5. Stochastic Oscillator: A momentum indicator comparing a particular closing price of an asset to a range of its prices over a certain period, with the sensitivity of the oscillator to market movements reduced by smoothing out the data.

6. Williams Percentage Range (%R): A momentum indicator that measures overbought and oversold levels, similar to a stochastic oscillator. It compares the close to the high-low range over a specific period.

7. Relative Strength Index (RSI): A momentum oscillator that measures the speed and change of price movements, indicating overbought or oversold conditions.

8. Moving Average Convergence/Divergence (MACD): A trend-following momentum indicator showing the relationship between two moving averages of a security's price.

9. MACD Signal: A signal line for the MACD, usually calculated as a 9-day EMA of the MACD values, which can be used as a trigger for buy and sell signals.

10. Bollinger Bands: A volatility indicator consisting of a middle SMA along with two standard deviation lines plotted above and below it to indicate overbought or oversold conditions.

11. On Balance Volume (OBV): A technical trading momentum indicator that uses volume flow to predict changes in stock price.

References

Dec 2023. URL `https://fred.stlouisfed.org/series/TB3MS`.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

Wei Chen, Haoyu Zhang, Mukesh Kumar Mehlawat, and Lifen Jia. Mean–variance portfolio optimization using machine learning-based stock price prediction. *Applied Soft Computing*, 100:106943, 2021.

Arsalan Dezhkam and Mohammad Taghi Manzuri. Forecasting stock market for an efficient portfolio by combining xgboost and hilbert–huang transform. *Engineering Applications of Artificial Intelligence*, 118:105626, 2023.

Frank J. Fabozzi, Harry M. Markowitz, and Francis Gupta. Portfolio selection. *Handbook of Finance*, 2008. doi: 10.1002/9780470404324.hof002001.

Adam Hayes. Treasury bills (t-bills): What you need to know to invest. URL `https://www.investopedia.com/terms/t/treasurybill.asp`.

Zhenlong Jiang, Ran Ji, and Kuo-Chu Chang. A machine learning integrated portfolio rebalance framework with risk-aversion adjustment. *Journal of Risk and Financial Management*, 13(7):155, 2020.

Li Jidong and Zhang Ran. Dynamic weighting multi factor stock selection strategy based on xgboost machine learning algorithm. In *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, pages 868–872. IEEE, 2018.

Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77, 1952. doi: 10.2307/2975974.

Noella Nazareth and Yeruva Yenkata Ramana Reddy. Financial applications of machine learning: A literature review. *Expert Systems with Applications*, page 119640, 2023.

Adeola Ogunleye and Qing-Guo Wang. Xgboost model for chronic kidney disease diagnosis. *IEEE/ACM transactions on computational biology and bioinformatics*, 17(6):2131–2140, 2019.

Nicola Uras and Marco Ortu. Investigation of blockchain cryptocurrencies' price movements through deep learning: a comparative analysis. In *2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 715–722. IEEE, 2021.

Mingxuan Wang. A portfolio strategy based on xgboost regression and monte carlo method. In *2022 2nd International Conference on Economic Development and Business Culture (ICEDBC 2022)*, pages 896–902. Atlantis Press, 2022.

Kyung Keun Yun, Sang Won Yoon, and Daehan Won. Prediction of stock price direction using a hybrid ga-xgboost algorithm with a three-stage feature engineering process. *Expert Systems with Applications*, 186:115716, 2021.