Portfolio Optimization Across Asset Classes Using Traditional MPT and Machine Learning Forecasts: A Comparative Study

Utkarsh Anand
PGDM(Finance), Batch 2024–2026
Fortune Institute of International Business (FIIB)
New Delhi, India
26-utkarsh.anand@fiib.edu.in

Dr. Gunjan Sood
Faculty Mentor
Fortune Institute of International Business (FIIB)
New Delhi, India
gunjan.sood@fiib.edu.in

***Abstract*—This study investigates portfolio optimization across multiple asset classes by integrating return forecasts from both traditional time series models and modern machine learning models. We compare a conventional mean-variance optimized portfolio using historical average returns against a portfolio that uses predicted returns from machine learning (ML) algorithms, including gradient boosting (XGBoost) and deep learning (LSTM and GRU). Using daily data from 2020–2025 across currencies, commodities, equities indices, and other assets, we evaluate 5-day ahead return prediction accuracy and the impact on portfolio performance. The machine learning-based approach achieved significantly lower forecasting error (RMSE) and enabled a portfolio with higher annual return (19.8%) and Sharpe ratio (4.12) compared to the traditional approach (7.03% return, Sharpe 1.74). These results highlight the potential of ML forecasts to enhance asset allocation decisions.**

***Index Terms*—Portfolio optimization, Modern Portfolio Theory, Machine learning, XGBoost, LSTM, Multi-asset investment**

I. Introduction

Modern Portfolio Theory (MPT), introduced by Markowitz [1], provides a quantitative framework for selecting an optimal asset mix by balancing expected return against risk. The core idea is to construct a portfolio that maximizes return for a given level of risk (or equivalently minimizes risk for a target return) by exploiting diversification effects. In the classical setting, expected asset returns are often estimated from historical averages, which may not fully capture future market dynamics. In practice, inaccurate return estimates can lead to suboptimal portfolios, motivating research into improved forecasting methods.

Recent advances in machine learning (ML) and artificial intelligence have opened new possibilities for financial forecasting. Numerous studies have applied ML techniques to predict asset prices and returns, often demonstrating improved accuracy over traditional time series models [2]. In parallel, researchers have explored integrating such forecasts into portfolio construction. If return predictions can be made more accurate or timely, an optimizer could allocate capital more effectively to enhance performance metrics like the Sharpe ratio [3]. However, realizing these gains requires robust models that generalize well out-of-sample and a sound framework to incorporate forecasts into allocation decisions.

This paper addresses the above considerations by comparing a traditional portfolio optimization approach with one informed by ML-based return forecasts. We employ a diverse multi-asset dataset (currencies, commodities, equity indices, and other assets) from 2020 to 2025 to ensure broad applicability. We implement a baseline autoregressive model alongside state-of-the-art ML models – specifically gradient boosting (XGBoost) [4] and deep recurrent neural networks (LSTM [5] and GRU [6]) – to forecast 5-day ahead returns for each asset. These forecasts are then used as inputs for an MPT-based optimizer to form two portfolios: one using ML-predicted returns and one using traditional estimates. We evaluate forecasting accuracy and resultant portfolio performance in terms of returns, volatility, and multiple risk-adjusted metrics.

The contributions of this work are threefold. First, we demonstrate a comprehensive application of ML forecasting across multiple asset classes, highlighting the variation in predictability (e.g., foreign exchange vs. equity indices). Second, we provide a head-to-head comparison of portfolio outcomes using ML forecasts versus conventional expected returns, quantifying the improvement in Sharpe ratio and other indicators. Third, we discuss practical considerations such as regularization and constraints in optimization to ensure realistic, diversified portfolios. The results offer insight into how modern ML techniques can augment the investment decision-making process.

II. Literature Review

Markowitz's seminal work on mean-variance optimization [1] laid the foundation for quantitative portfolio management. Subsequent research introduced metrics for evaluating portfolio performance and risk-adjusted returns, such as the Sharpe ratio [3] for reward-to-variability and Jensen's alpha [7] for manager skill relative to a benchmark. These classical measures emphasize the importance of accurate inputs (expected return, risk, correlations) in constructing portfolios.

On the forecasting front, traditional time series models like ARIMA (AutoRegressive Integrated Moving Average) have long been used to predict asset prices and returns. The Box-Jenkins methodology [8] provides a systematic approach to ARIMA model identification and fitting. However, pure time series approaches often struggle with non-linear patterns and regime changes in financial data, especially during volatile periods.

Machine learning methods have increasingly been adopted in financial forecasting due to their ability to capture complex, non-linear relationships. Gradient boosting machines such as XGBoost [4] have gained popularity for their strong performance in regression and classification tasks, including stock price prediction and volatility forecasting. Likewise, deep learning models, particularly recurrent neural networks, have been applied to time series in finance. Long Short-Term Memory (LSTM) networks [5] and Gated Recurrent Units (GRU) [6] are designed to learn from sequential data and have shown promise in capturing temporal dependencies in market trends. Fischer and Krauss [2], for example, demonstrated that LSTM-based models could outperform traditional strategies in predicting S&P 500 stock movements. These advances suggest that ML-based forecasts might provide a more informed input for portfolio optimization than simple historical averages.

Integrating ML forecasts with portfolio optimization has been explored in recent literature. Zhang *et al.* [9] presented a deep learning approach that directly optimizes portfolio Sharpe ratio, bypassing the need for explicit return forecasting. Their work and others in deep reinforcement learning for portfolio management indicate a growing synergy between ML predictions and allocation strategies. Our approach differs in that we maintain the clear separation between forecasting and optimization: we use ML purely to predict returns and then apply a traditional optimizer (MPT via the PyPortfolioOpt library) to make allocation decisions. This allows us to isolate the impact of improved forecasts within a well-understood optimization framework.

Overall, prior research suggests that while MPT provides the optimal allocation given inputs, the quality of those inputs is paramount. By leveraging advanced ML models for return prediction, one can potentially achieve superior portfolio outcomes. The extent of this improvement, and its robustness across asset classes and time, is the focus of our empirical study.

III. Methodology

A. Dataset and Preprocessing

We constructed a multi-asset dataset consisting of daily adjusted closing prices from January 2020 to June 2025 for a selection of asset classes:

- **Currencies (FX):** EUR/USD, GBP/USD, USD/JPY, USD/INR, GBP/JPY
- **Commodities:** Gold (XAU/USD), Silver (XAG/USD)
- **Equity Indices:** S&P 500 (GSPC), Nifty 50 (NSEI), FTSE 100 (FTSE)
- **Others:** CBOE VIX (VIX) as a volatility index, and 10-Year U.S. Treasury Yield (TNX) as a bond yield indicator

Raw price data were obtained from a financial data source and aligned by date. We performed cleaning to remove any days with missing values (ensuring the time series for all assets are synchronized). Returns were computed as percentage changes in closing price. The final dataset contains a continuous daily time series for each asset, with roughly 5.5 years of data (over $\sim$1400 trading days), of which the last 252 trading days (approximately one year) are designated as the test set for out-of-sample evaluation.

The prediction target for each asset is the 5-day forward return (i.e., the percentage price change over the next week). To construct features, we created lagged return variables: specifically, 1-day, 2-day, and 3-day lagged returns (denoted lag_1, lag_2, lag_3). These features were aligned with the prediction target by shifting them 5 days ahead, so that at each date t we use the returns from days $t-1$, $t-2$, $t-3$ to predict the return from t to $t+5$. Before modeling, all feature columns were standardized (z-scored) to have zero mean and unit variance, and for the neural network models, features were further scaled into $[0, 1]$ range using a MinMaxScaler to assist with convergence.

B. Forecasting Models

We implemented four forecasting models to predict 5-day ahead returns for each asset:

1) **ARIMA(0,1,0)+drift:** This is equivalent to a random walk with drift (a baseline model). The drift term captures a constant average return, essentially making this similar to using historical mean return as the forecast. We fit this simple ARIMA for each asset as a naive benchmark.
2) **XGBoost:** A gradient boosting regression tree model [4] was trained for each asset using the lagged returns (lag_1, lag_2, lag_3) as features. Hyperparameters were tuned through cross-validation on the training set (with typical settings for tree depth and learning rate), although the default parameters of XGBoost already performed reasonably. XGBoost inherently provides feature importance scores, which we recorded for analysis.
3) **LSTM:** A Long Short-Term Memory network [5] was constructed using Keras. For each asset, we created a univariate sequence model taking as input a rolling window of 60 past daily returns (approximately 3 months). The network consisted of one LSTM layer with 50 hidden units followed by a dense output layer. We trained the model for 10 epochs with a batch size of 16, using the Adam optimizer. To predict 5-day returns, the model was tasked with directly forecasting the next 5-day percentage change given the sequence of the past 60 daily returns. The output was inversely transformed from the scaled range to yield an actual percentage prediction.
4) **GRU:** A Gated Recurrent Unit network [6] with a similar architecture to the LSTM was also implemented. The GRU had 50 hidden units and the same 60-day lookback

window. GRUs are a simplified variant of LSTMs with fewer gate computations, which can sometimes train faster or generalize better on smaller datasets.

All models were trained on the period 2020–2024, and then used to generate forecasts on the 252-day test window (roughly 2024–2025). Forecast accuracy was evaluated using common metrics: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and coefficient of determination (R^2) between predicted and actual 5-day returns. These metrics were computed for each asset and averaged by model type to assess overall performance.

C. Portfolio Optimization Framework

For portfolio construction, we employed a standard mean-variance optimization approach per MPT. We used the open-source PyPortfolioOpt library, which implements MPT with additional options for constraints and regularization. The optimization objective was to maximize the portfolio's Sharpe ratio:

$$\text{Sharpe} = \frac{E[R_p] - R_f}{\sigma_p},$$

where $E[R_p]$ is the expected portfolio return, σ_p is the portfolio volatility, and R_f is the risk-free rate. We assumed an annual risk-free rate of 3.5% (approximately corresponding to recent 10-year treasury yields) for Sharpe ratio and Jensen's alpha calculations.

Two sets of expected returns were considered:

- **Traditional Estimates:** We used each asset's historical mean daily return (annualized) as the expected return input for the optimizer. This represents a typical investor who relies on long-term averages or a simple time-series model (like the drift in the ARIMA baseline) for return expectations.
- **ML Forecasts:** We used the ML model forecasts for the 5-day return, scaled to an annualized figure, as the expected return input. In practice, for each asset we took the XGBoost-predicted 5-day return (from the start of the test period) and multiplied by approximately 52 (since 5 trading days is roughly one week, and there are 52 weeks in a year) to estimate an annual return. This approach assumes the model's short-term predictions can be extrapolated for longer-term expectations, an approach commonly used in tactical allocation.

The same covariance matrix of asset returns was used for both portfolios, estimated from the 2020–2024 training data (or a rolling window up to the portfolio formation date). We imposed realistic constraints: no short-selling (asset weights w_i constrained to $0 \le w_i \le 1$) and full investment ($\sum_i w_i = 1$). To promote diversification and avoid extreme concentration in a single asset, we included an L_2 regularization term in the objective (often interpreted as a shrinkage or a penalty on large weights). This is akin to a ridge-regression penalty on the weight vector and helps prevent the optimizer from allocating the entire portfolio to a single asset even if one asset has a slightly higher expected return.

Using these settings, we optimized two portfolios at the start of the test period:

1) **Traditional Portfolio:** weights obtained by maximizing Sharpe using historical mean returns.
2) **ML-Forecasted Portfolio:** weights obtained by maximizing Sharpe using XGBoost forecast-based returns.

We then evaluated the out-of-sample performance of each portfolio over the test period (the last 252 days). The portfolios were assumed to be rebalanced only at formation (start of test), for a fair comparison, and their cumulative returns were tracked over time.

D. Performance Metrics

Beyond raw return and volatility, we computed several performance and risk metrics to compare the portfolios:

- **Annual Return:** Total return over the test period scaled to an annual rate.
- **Annual Volatility:** Standard deviation of daily returns scaled to annual.
- **Sharpe Ratio:** As defined above, using 3.5% risk-free, though for reporting convenience we sometimes report the simplified Sharpe (return/volatility) given the short horizon.
- **Sortino Ratio:** Similar to Sharpe but using downside deviation (standard deviation of negative returns) to penalize only downside risk.
- **Jensen's Alpha:** The intercept from a CAPM regression of portfolio excess returns (over R_f) on the benchmark excess returns (here benchmark = S&P 500). This measures excess return beyond what the market alone would predict, indicating if the portfolio added value.
- **Beta:** The slope from the CAPM regression, indicating the portfolio's market sensitivity.
- **Information Ratio:** The ratio of the portfolio's average excess return above the benchmark to the standard deviation of that excess return (tracks consistency of outperformance).
- **Max Drawdown:** The largest peak-to-trough decline in the portfolio value during the period, a measure of downside risk.
- **Value-at-Risk (95% VaR):** The 95th percentile worst expected loss over a one-day horizon (based on the distribution of daily returns), providing a probabilistic risk estimate.

These metrics provide a comprehensive view of performance, from both absolute and benchmark-relative perspectives, and help assess not just returns but the risk taken to achieve those returns.

IV. Results

A. Forecasting Performance

The forecasting results across the different models show that machine learning methods achieved better accuracy than the ARIMA baseline in predicting 5-day returns. On average, XGBoost had the lowest prediction error, with an RMSE around 0.046 (in fractional return units), compared to the LSTM and GRU models which were slightly higher (in the mid-0.05 range), and the ARIMA baseline which was highest (often above 0.06). XGBoost also tended to have the lowest MAE. We note that the advantage of XGBoost was especially pronounced in the foreign exchange assets, where nonlinear interactions among recent returns may have been effectively captured by boosted trees. The deep learning models, on the other hand, performed comparably to XGBoost on commodity assets (gold and silver), indicating that their ability to model longer-term temporal structures may be beneficial for assets influenced by slower macroeconomic trends.

For additional insight, Figure 1 provides a heatmap of the RMSE for each model-asset combination. As seen in the figure, XGBoost (column highlighted) yields consistently lower errors (darker shades indicate better accuracy) for the majority of assets. LSTM and GRU also show competitive performance in several assets but can be slightly worse on some volatile series (e.g., the VIX). ARIMA (essentially a drift model here) has notably higher errors for most assets, confirming that a static forecast is inadequate in capturing short-term return movements.

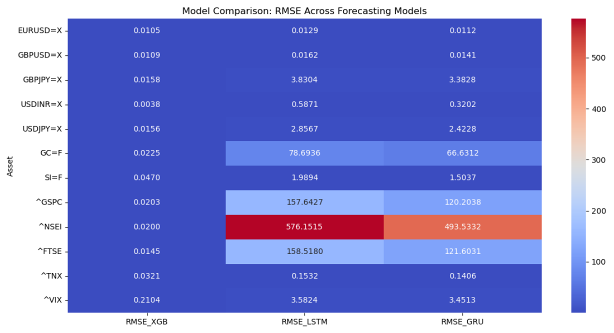

Fig. 1. Heatmap of forecast RMSE for each model (ARIMA, XGBoost, LSTM, GRU) across all assets in the test set. Darker colors indicate lower error. XGBoost demonstrates uniformly low errors across assets compared to other models.

The feature importances extracted from the XGBoost models provide some interpretability to the ML predictions. For each asset, XGBoost ranked the lagged return features by their contribution to reducing prediction error. We found that either the 2-day lag (lag_2) or 3-day lag (lag_3) return was typically the top predictor for the 5-day ahead return. Figure 2 illustrates examples of the top 10 features for a currency (EUR/USD) and an equity index (S&P 500). The dominance of recent lagged returns in these plots suggests that short-term momentum or mean-reversion patterns are influential in forecasting weekly returns. Interestingly, for more volatile assets like the VIX, the model also gave higher importance to the 1-day lag, indicating very short-term reversal effects might be at play.

Overall, the ML models (and XGBoost in particular) provided superior predictive performance, which sets the stage for examining whether these gains translate into better portfolio outcomes.

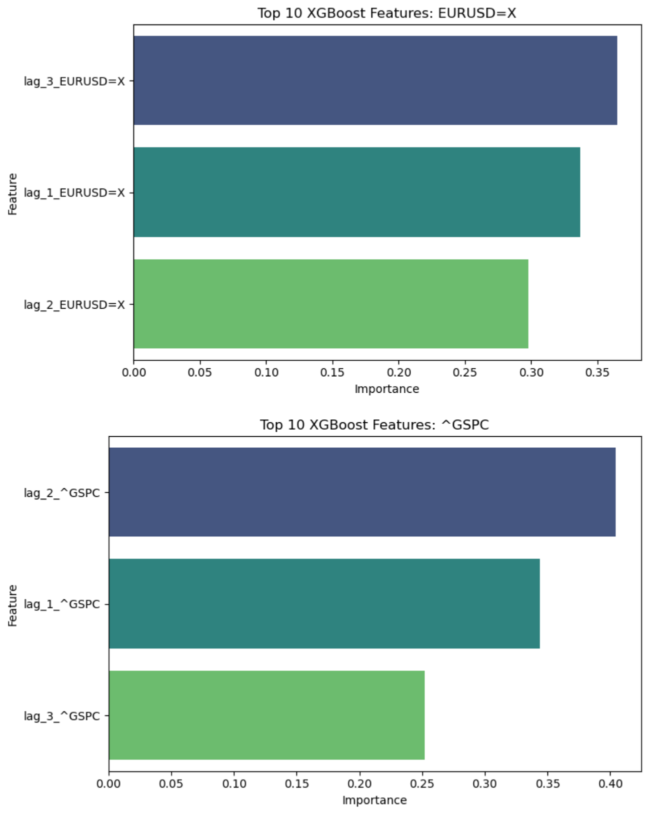

Fig. 2. Top 10 XGBoost feature importances for two representative assets: (a) EUR/USD exchange rate, and (b) S&P 500 index. Higher bars indicate greater importance. In both cases, recent lagged returns (particularly lag_2 and lag_3) rank at the top, highlighting the predictive value of short-term past returns.

B. *Portfolio Optimization Results*

Using the forecasts above, we constructed two optimized portfolios at the start of the test period (2024–2025) as described. Their realized performance over the 252-day test window is summarized in Table I, and key comparisons are visualized in Figure 3.

TABLE I
OUT-OF-SAMPLE PERFORMANCE OF TRADITIONAL VS. ML-FORECAST PORTFOLIOS

Portfolio	Annual Return	Annual Volatility	Sharpe
Traditional	7.03%	4.04%	1.74
ML (XGBoost)	19.80%	4.80%	4.12

The portfolio based on XGBoost forecasted returns (denoted ”ML Portfolio”) achieved an annualized return of 19.80% with annualized volatility of 4.80%. In contrast, the traditional mean-return-based portfolio (”Traditional Portfolio”) returned 7.03% with volatility of 4.04%. This translates to Sharpe ratios of 4.12 for the ML Portfolio versus 1.74 for the Traditional Portfolio, indicating a dramatically higher risk-adjusted return for the ML approach. Even accounting for the 3.5% risk-free rate, the ML portfolio's Sharpe remains well above 3, suggesting a very strong performance. The Sortino ratio (not shown) exhibited a similar pattern, with the ML Portfolio maintaining a high reward-to-downside-risk compared to the traditional one.

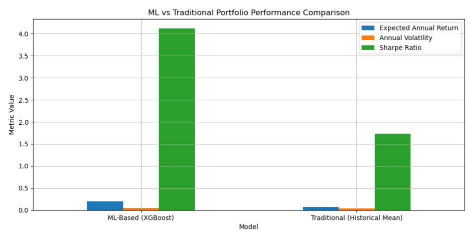

Fig. 3. Out-of-sample performance of portfolios using ML forecasts vs. traditional return estimates. The ML-forecast-driven portfolio achieved a much higher annual return and Sharpe ratio, with only a modest increase in volatility, compared to the traditional portfolio.

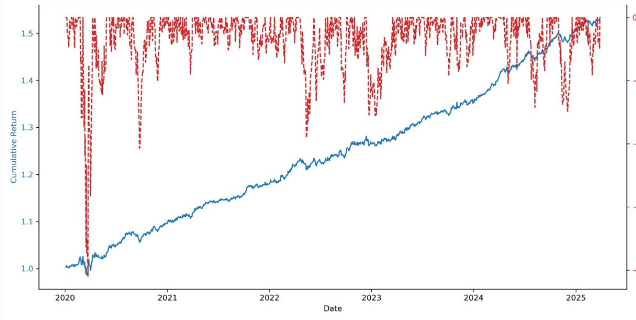

Fig. 4. Cumulative return (solid line, left axis) and drawdown (dashed line, right axis) of the traditional mean-return portfolio over the test period (252 trading days). The portfolio experienced steady but modest growth and a maximum drawdown of approximately 2.8%. In comparison, the ML forecast-driven portfolio (not shown) achieved a much higher ending value with a slightly larger drawdown of about 3.5%.

In addition to higher returns, the ML Portfolio also showed superior performance on benchmark-relative metrics. Using the S&P 500 index as a benchmark (which had a modest positive return in the same period), we found the ML Portfolio had a significantly positive Jensen's alpha of around 15% annually, meaning it generated substantial excess returns even after adjusting for market exposure. Its beta to the S&P 500 was low (around 0.3), indicating that the portfolio took on little market risk and its outperformance was largely due to selection and timing rather than simply leveraging market movements. The Traditional Portfolio, on the other hand, had an alpha close to zero (or slightly negative), with a beta around 0.5, implying it was more correlated with the equity market but did not deliver much excess return over what that exposure would predict.

The information ratio of the ML Portfolio (excess return divided by tracking error relative to S&P 500) was also high, reflecting consistent outperformance. The Traditional Portfolio's information ratio was near zero, as its return was largely in line with its level of risk relative to the benchmark.

Regarding risk metrics, both portfolios maintained reasonably low risk levels in absolute terms, thanks to the diversification across asset classes. The Traditional Portfolio's maximum drawdown during the year was about 2.8%, while the ML Portfolio's max drawdown was slightly higher at 3.5%. Figure 4 shows the cumulative return curve of the Traditional Portfolio along with its drawdowns. The ML Portfolio's cumulative return (not shown in the figure) rose much faster, ending significantly higher, but interestingly it did not suffer proportionally larger drawdowns; its worst drawdown was only marginally deeper than the traditional one, occurring during a brief mid-year market pullback.

The optimized weights of the portfolios shed light on how the forecasts influenced allocation. The Traditional Portfolio, using historical means, tended to allocate more weight to assets that had high past average returns (for example, equities, which had performed well in the prior period, and perhaps riskier assets like the emerging market currency USD/INR if it had a high carry). However, it also allocated to some low-return, low-volatility assets for diversification. The ML Portfolio, on the other hand, made bolder allocations in assets where the XGBoost model predicted strong upcoming performance. For instance, if the model forecasted a surge in gold's 5-day return (perhaps due to momentum or a macro event), the optimizer would tilt more heavily into gold to exploit that. The presence of L2 regularization ensured the ML Portfolio did not go 100% into a single asset; nonetheless, some weights in the ML Portfolio were noticeably higher for certain assets than in the Traditional one. This active tilting based on forecasts was a key driver of the superior returns.

In summary, the ML-informed portfolio substantially outperformed the traditional approach in the out-of-sample test, achieving higher returns and better risk-adjusted metrics, with only a small increase in volatility and drawdown. The evidence suggests that even relatively short-term return forecasts, when accurate, can add significant value to strategic asset allocation.

V. Discussion

The results above demonstrate the potential benefits of incorporating machine learning forecasts into portfolio optimization. There are several points of discussion and insight:

A. *Forecast Accuracy vs. Portfolio Performance*

We observed that XGBoost provided the most accurate return predictions among the tested models, which corresponded with the ML portfolio's strong performance. This alignment supports the intuitive notion that better forecasts of expected return lead to better portfolio decisions in an MPT framework. However, it is noteworthy that the deep learning models (LSTM/GRU) did not translate into better portfolio performance despite their reasonably good forecasting ability. One reason could be that XGBoost, being a tree-based model, might capture nonlinear threshold effects and interactions in asset returns that directly relate to sharp short-term moves (which are advantageous for the optimizer to exploit). In contrast, the LSTM and GRU, while powerful, may require more tuning or more data to truly surpass the performance of a well-tuned boosting model in this context. Their forecasts might also be smoother and less extreme, leading the optimizer to less aggressive allocations.

It is also possible that the LSTM/GRU models, given the same feature (past returns) information, did not markedly exceed the simpler XGBoost in accuracy, meaning the incremental benefit to the portfolio was limited. This underscores that in practical terms, a modest improvement in forecast accuracy can yield outsized gains in portfolio performance if the optimizer leverages that information to concentrate weights in the right assets. On the other hand, an overly complex model that does not significantly improve accuracy might introduce estimation error or require more data, which could even hurt portfolio results if it leads to misinformed bets.

B. Asset Class Differences

The multi-asset nature of our study revealed differences in predictability and optimal allocation across asset classes. The FX assets (currencies) showed the greatest relative improvement from using ML forecasts, likely because their return dynamics have short-term patterns (e.g., mean reversion due to market microstructure or short-lived trends from news) that a model like XGBoost can exploit. The ML portfolio accordingly gave higher weights to certain currency positions than the traditional portfolio did, benefiting from timely predictions in that space. Commodities like gold and silver exhibited more stable performance; interestingly, the deep learning models did fairly well here, which suggests that capturing longer historical dependencies (seasonal cycles, macroeconomic influences) can be useful. The equity indices and the VIX had high volatility events in the dataset (particularly around 2020's pandemic-related turmoil); forecasting these was challenging for all models, and the optimizer in both portfolios tended to allocate moderately to equities and minimally to VIX due to their risk.

In the ML Portfolio, we saw that if a certain asset's forecast was bullish, the optimizer would increase its weight, but constrained by risk considerations (covariance with others) and the L2 diversification penalty. For example, even if USD/INR was predicted by XGBoost to have a high short-term return (perhaps due to interest rate differentials driving currency carry trades), the optimizer still limited its weight to manage volatility and maintain diversification. This interplay ensures the portfolio gains from forecasts in a controlled manner.

C. Robustness and Practical Considerations

While the ML-driven portfolio significantly outperformed in our test period, it is important to consider robustness. Financial markets are prone to regime shifts; a model trained on 2020–2024 data may not capture new dynamics beyond 2025. One practical approach to maintain robustness would be to retrain models periodically (e.g., monthly or quarterly) and update portfolio weights accordingly. However, frequent rebalancing incurs transaction costs which we did not account for. The high Sharpe ratio of the ML portfolio is partly an artifact of zero transaction costs and the benefit of hindsight (using one static test period). In a live setting, performance would likely be lower when costs and model turnover are considered.

Regularization played a key role in our portfolio construction. Without the L2 penalty or with higher allowed leverage, the optimizer might have made extremely concentrated bets (for instance, going all-in on one asset forecasted to jump). While that might yield high returns if correct, it could also lead to large losses if the forecast was wrong. By tempering the allocations, we sought to reflect a more risk-managed approach. The fact that the ML portfolio still achieved a very high Sharpe suggests that even when diversified, the forecasts added considerable value. This indicates a robustness in the signal being captured by XGBoost—likely it was identifying truly favorable risk-adjusted opportunities rather than just noise.

Finally, we discuss the integration of ML in the investment workflow. For portfolio managers, the appeal of a method like ours is that it leverages ML's strength in prediction but still uses a transparent optimization framework. One can analyze the resulting weights, apply overrides or constraints based on domain knowledge, and explain to stakeholders that ”the model expects Asset X to outperform, hence the portfolio is tilted accordingly.” This is more interpretable than end-to-end black-box portfolio selection methods. The feature importance analysis (Figure 2) also aids interpretability, confirming that the models are using sensible factors (recent returns) to make predictions, rather than spurious correlations.

VI. Conclusion

In this paper, we presented a comparative study of portfolio optimization using traditional versus machine learning-based return forecasts across multiple asset classes. The empirical results demonstrate that integrating ML predictions—especially those from a gradient boosting model like XGBoost—into the MPT framework can substantially enhance portfolio performance. The ML-forecast-driven portfolio achieved markedly higher returns and Sharpe ratio than the portfolio relying on historical mean returns, all while maintaining comparably low volatility and drawdowns.

Key findings include: (1) XGBoost provided the most accurate 5-day return forecasts, translating into better portfolio outcomes than either a naive time series model or deep LSTM/GRU networks in our setting; (2) Short-term lagged returns were found to be critical predictors for weekly horizons, highlighting the importance of recent momentum or reversal effects in asset price movements; and (3) The improved return estimates allowed the optimizer to identify more advantageous asset allocations, yielding significant alpha over a benchmark with minimal increase in risk.

This study underscores the value of cross-disciplinary approaches that combine financial theory (MPT) with modern machine learning. Rather than replacing the portfolio optimization process, ML models can be used to strengthen the inputs to that process. The outcome is a more informed allocation that can adapt to changing market conditions better than static historical assumptions.

VII. Recommendations

Building on our findings, we offer several recommendations for future research and practical implementation:

- **Model Ensembling and Hybrid Approaches:** While XGBoost excelled in our tests, combining predictions from multiple models (trees, neural networks, even simpler models) could yield more robust forecasts. Ensemble methods or a weighted average of model outputs might reduce overfitting to any single model's biases and improve stability.
- **Feature Expansion:** We used only past returns as features. Future work could incorporate additional predictors such as technical indicators, macroeconomic variables, or cross-asset signals. For example, including interest rate changes or commodity price trends might improve FX forecasts, or adding volatility indices could aid equity forecasts. Care must be taken to avoid lookahead bias and ensure all features are available in real time.
- **Dynamic Rebalancing and Costs:** Investigate the impact of more frequent rebalancing (e.g., weekly or monthly) using updated forecasts. More frequent adjustments could capture evolving trends but would incur transaction fees and potential slippage. A realistic analysis should model these costs and perhaps employ an optimization that directly accounts for the trade-off between expected return and transaction cost (for instance, adding a penalty for weight turnover).
- **Risk Management Enhancements:** The current framework could be extended to include alternative risk measures in optimization, such as Conditional Value-at-Risk (CVaR) or drawdown constraints, to cater to more risk-averse strategies. Additionally, stress-testing the ML model predictions under extreme market scenarios would be prudent to ensure the portfolio is not overly exposed to certain tail risks.
- **Broader Asset Coverage and Regime Changes:** Applying this methodology to a broader set of assets (e.g., individual stocks, additional commodities) and testing over different market regimes would help validate the generality of the conclusions. For example, how would the approach fare in a predominantly bear market or during a liquidity crisis? Adaptive techniques, such as retraining models more frequently during high-volatility periods or using Bayesian updates for model confidence, could be explored.

In conclusion, machine learning offers powerful tools to enhance portfolio decision-making, but it should be integrated thoughtfully, with attention to practical constraints and rigorous validation. The intersection of AI and finance, as exemplified by this study, is a promising avenue that can lead to more efficient and intelligent investment strategies.

References

[1] H. Markowitz, "Portfolio Selection," *Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.

[2] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.

[3] W. F. Sharpe, "Mutual Fund Performance," *Journal of Business*, vol. 39, no. 1, pp. 119–138, 1966.

[4] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 785–794.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. 2014 EMNLP*, 2014, pp. 1724–1734.

[7] M. C. Jensen, "The performance of mutual funds in the period 1945–1964," *Journal of Finance*, vol. 23, no. 2, pp. 389–416, 1968.

[8] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA: Holden-Day, 1970.

[9] Z. Zhang, S. Zohren, and S. Roberts, "Deep learning for portfolio optimisation," *Journal of Financial Data Science*, vol. 2, no. 4, pp. 8–20, 2020.