Volume 39(2), pp. 143–157
http://orion.journals.ac.za

ORiON
ISSN 0259–191X (print)
ISSN 2224–0004 (online)

A hybrid predictive prototype for portfolio selection using probability-based quadratic programming and ensemble artificial neural networks

Brian Muganda, * and Bernard Shibwabo*

Received: 8 Augustus 2022; Revised: 31 January 2023; Accepted: 24 February 2023

Abstract

Often, investors are limited by cognitive and emotional biases in their decision making which leads to poor portfolio investment choices. Robo-advisors can assist in overcoming these biases. This paper seeks to develop a financial robo-advisor prototype based on hybrid programming. It uses ensemble artificial neural networks to predict portfolio returns and variances with input nodes of Ornstein–Uhlenbeck processes (OU) and geometric Brownian motion (GBM) processes' estimates. The results are subsequently channeled into a probability quadratic optimization algorithm which considers target return probability and value-at-risk constraints as proxies for investor's risk tolerance so as to provide the optimal portfolio allocation strategy that minimizes portfolio risk given a prescribed investment horizon and target return. The results show that the ensemble artificial neural network method implementation accurately predicted the level of 2 of 5 assets and the trends of the remaining 3 assets. However, it yielded low standard deviations and low returns compared to the OU and GBM estimates for short horizons. The quadratic optimization algorithm supported investment in shorter time horizons since portfolio risk was lowest. Diversified allocation was achieved in the shorter time horizons while longer horizon allocations were biased towards assets with lower standard deviations. The lowest risk portfolios were the ones with a lower certainty probability for the target return and vice versa. This paper clearly demonstrates that ensemble methods are accurate in prediction, and that a hybrid programming paradigm effectively leverages the strengths, speed and functionality of different programming languages — an elixir for multifaceted dissociable programming problems.

Key words: Artificial neural networks; Ensemble methods; Prediction; Certainty probability constraint; Portfolio selection; Value-at-risk; Quadratic programming.

*Business and Management Studies, Strathmore University, Kenya
email: brian.muganda@tukenya.ac.ke

http://dx.doi.org/10.5784/39-2-002

1 Introduction

A portfolio is a collection of investments held by an investment company, hedge fund, financial institution or an individual. This collection of investment features a combination of financial assets such as stocks, bonds or options. The holder of the portfolio can entrust its management to a financial professional who works on their behalf at an agreed management fee. The design of a portfolio is done according to the investor's risk tolerance, investment timeframe and investment objectives. Investors have varying risk preferences, investment timeframes and investment objectives, thus a variety of methods may be applied to design a portfolio that has proper fund allocation to each asset and select assets that meet the investor's objectives ([11]; [13]; [19]).

A robo-advisor is an automated investment platform that uses quantitative algorithms to manage investors' portfolios and at the same time, is accessible to clients. Robo-advisors provide automated financial planning services with little to no human supervision. They digitally onboard clients while collecting information about them, their financial situation and future investment objectives. After assessing the client's risk profile, it uses this data, combined with market data, to offer advice and/or automatically allocate and rebalance the client's portfolios. A robo-advisor can also be used to support portfolio managers in their decision-making processes, providing them with 'smart' and automated recommendations, alerts and dashboards ([18]; [13]). This approach contrasts considerably with the relatively opaque, expensive, and non-digital securities products offered by traditional banks. Due to the high degree of automation and the portfolio-mapping, robo-advisors can offer their services within considerably better developed conditions than traditional asset management companies [15].

First- and second-generation robo-advisory systems comprise of online questionnaires and online proposals, thereby providing a combination of advice and online access to traditional manual asset management services. In contrast, the third and fourth generations of robo-advisory systems use quantitative methods and algorithms to construct and rebalance portfolios, thereby performing truly automated portfolio management. They cover the entire investment and portfolio management process, starting from the selection of the asset portfolio and conclude with periodic portfolio rebalancing and appropriate performance reporting [5].

Robo-advisory services can be offered at much lower costs as compared to traditional human advisors. These services also feature transparent workflows and monitoring systems, require a low or even no minimum investment, and utilize advanced quantitative methods of portfolio management and optimization. They can also achieve approximately the same returns on investments as traditional advisors. Users of robo-advisors can have access to various options to control, customize, and construct investment portfolios from multiple devices such smartphones or laptops [2].

In 2017, the global assets under management by robo-advisors comprised approximately \$226 billion, and the number of users of these robo-advisor systems exceeded 12 million. A forecast by [19] predicts that RAs would manage about \$1.78 trillion globally; moreover, assets under management are expected to show annual growth rate of 15.5% between 2022- 2026, with a user base of about 510 million people by 2026. The main reasons for the growing success of robo-advisory systems can be attributed to this new generation of clients and the advantages of these systems over traditional financial advisors. The success concerning robo-advisors has been attributed to transparent, cost-effective and technological offerings. In order to maximize the chances of success, the banks must deal with the individual, institutional and technical challenges of robo-advisor implementation and create the strategic or organizational framework for the digitalization of securities and investment advising ([15]; [19]; [3]).

In their study, [2] analyzed 219 existing robo-advisors worldwide and found that Modern Portfolio Theory remains the main framework used in these robo-advisors. They found that the current trend is to attempt to improve and augment this framework rather than develop entirely new approaches. The asset allocation methodology adopted by 80% of robo-advisors is the Modern Portfolio Theory approach, supplemented by about 8% with the Black–Litterman, VaR and Conditional VaR optimization techniques. Meanwhile, 20% adopted Monte Carlo simulation approaches.

In their review, [13] opine that investors using mean-variance optimization may reduce the effects of estimation error by applying reasonable constraints, conducting sensitivity analysis, performing robust optimization, or using the Black-Litterman model. They, however, note that some of the solutions are not mutually exclusive. They further suggest that investors wishing to use the mean-variance optimization technique should consider further simulations that perturb the underlying parameters, including the mean, standard deviations, correlations, and value-at-risk (VaR) coefficients.

In a study by [7] on portfolio selection under the Markowitz model using expected returns prediction applying neural networks, the results show that the neural network predictor, which uses the autoregressive model with a lag parameter of 4 as the training model, can produce better estimates for future returns than the historical time series mean value. In his study, [6] also investigates a neural networks approach for portfolio selection where he suggests that if the relationship between the output and input variables is non-linear, the linear regression model may not be a suitable choice for the prediction of the returns. He proposes the use of an Artificial Neural Network (ANN) which can capture non-linearity. Furthermore, [10] suggests that practical and theoretical limitations of the mean-variance model have led to the proposal of different utility functions, risk measures, and dynamic multi-period models that allow rebalancing of the portfolio to hedge against adverse market conditions. He adds that new legislation has often resulted in the necessity to introduce new classes of constraints on the portfolio composition.

Clients investing using robo-advisors typically expect that their investments will be managed with advanced, scientifically justified, modern, and well-implemented methods and technology. However, there currently exists a gap between the predominant methods applied in robo-advisors and new methodological developments. Therefore, this paper aims to develop a robo-financial advisor prototype which adopts a hybrid programming architecture and uses ensemble machine learning techniques to provide an optimal portfolio allocation strategy that maximizes investor returns for the prescribed investment horizons, budget constraints and risk levels. The prototype utilizes an ensemble learning technique of weighting prediction algorithms by using artificial neural networks to predict the expected price of assets in the portfolio. The underlying inputs in the prediction algorithm are the multi-asset OU and GBM models.

An ensemble modeling technique allows for multiple diverse base models to be used to predict outcomes. The motivation for using an ensemble modelling technique in the prediction is to reduce the generalization error. This ensemble model, which is adopted by using an artificial neural network with base predictors being OU and GBM models, acts and performs as a single prediction model. The combination of multiple models generally improves the accuracy of the prediction and works better than a single model [12]. A probability-based quadratic optimization algorithm is then applied to provide the optimal portfolio allocation strategy that maximizes an investor's returns in the given investment horizon, all within a hybrid programming architecture. The use of hybrid programming combines different programming language paradigms in a single application and provides the functionality of utilizing a rich set of core libraries cross-functionally within the application, thereby improving system performance.

2 Materials and methods

This study adopts a quantitative design approach to develop a probability-based quadratic optimization model for a hybrid portfolio selection prototype. Artificial neural networks are applied for multi-asset prediction, with the input models being of OU and GBM process types. The prototype is tested by creating a portfolio of 5 assets. The selected sample of assets consists of Exchange Traded Funds (ETFs) and Index Linked Funds assets, namely the SPDR Dow Jones REIT ETF (RWR), iShares Russell 2000 ETF (IWM), S&P 500 (GSPC), iShares MSCI EAFE ETF (EFA) and iShares MSCI Frontier 100 ETF (FM). This data is obtained from Eikon. The development environment of the prototype is a multilanguage programming paradigm that interfaces Python and R to leverage the speed and libraries of these respective programming languages. Figure 1 summarizes how the algorithms are developed in a hybrid computing environment with the base algorithms of prediction using artificial neural networks and quadratic optimization algorithms executed in R, while the development of the prototype is done within Python to leverage on Python's speed and application development utilities. In Figure 1, the hybrid programming architecture is demonstrated. The prototype is developed in a hybrid programming environment where the application development is done in Python while the artificial neural network and quadratic optimization algorithms for portfolio allocation are deployed in R but within the Python application development. The inputs for both algorithms are also presented – the actual stock data is required so as to estimate the OU and GBM models and also to train the neural network for the asset price predictions. The probability of target return and value-at-risk level, durations and the target returns are also required as inputs of the quadratic optimization algorithm so as to obtain the portfolio allocation weights. These have been presented in Figure 3.

The approach to finding the optimal portfolio allocations considers a probability/value-at-risk approach incorporated into the quadratic optimization problem where the expected stock return and variance forecasts are obtained using a neural network that has four hidden layers, a linear activation function and multi-asset OU and GBM process estimates as neural network inputs. Particularly, it considers that the daily return R_t on an individual

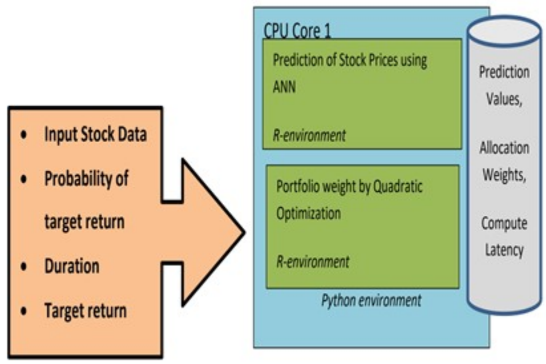

Figure 1: *Hybrid programming architecture, showing ANN prediction, optimization and prototype development environments.*

stock, S_t, at time t is defined as

$$R_t = \frac{S_t - S_{t-1}}{S_{t-1}}. \tag{1}$$

The portfolio return, R_p, therefore, is the sum of all the individual stock returns for a given holding period, h, and the allocation weights, w_i, represent the proportion of funds invested in the i^{th} asset in the portfolio with a total of N assets. This portfolio return for the given holding period is obtained by R_p^h as:

$$R_p^h = \sum_{i=1}^{N} w_i R_i^h. \tag{2}$$

The expected portfolio return, $E(R_p)$ is

$$E(R_p) = \sum_{i=1}^{N} w_i E(R_i).$$

The target portfolio return, $\hat{R}_p$, is the return that the investor desires to achieve. Given some investor certainty probability, β, of a achieving this portfolio return, and by assuming normality of the asset returns, the target return, $\hat{R}_p$, is obtained by

$$\hat{R}_p = E(R_p) + \phi^{-1}(\beta)\sigma_p, \tag{3}$$

where $\phi^{-1}(\beta)$ is the quantile function of the standard normal distribution evaluated at β, and σ_p is the standard deviation of the portfolio returns. Further, a condition is set that the required value-at-risk (total portfolio loss) should be at most zero as presented in (4). This implies that given only α probability level of loss, then the loss on the portfolio would be at least zero. By this, we then have that

$$0 = E(R_p) + \phi^{-1}(\alpha)\sigma_p. \tag{4}$$

Thus from (3) and (4), the target return can be expressed as

$$\begin{aligned}\hat{R}_p &= E(R_p) - E(R_p)\frac{\phi^{-1}(\beta)}{\phi^{-1}(\alpha)} \\ &= \left(1 - \frac{\phi(-1)(\beta)}{\phi(-1)(\alpha)}\right)E(R_p) \\ &= \left(1 - \frac{\phi^{-1}(\beta)}{\phi^{-1}(\alpha)}\right)\sum_{i=1}^{N} w_i E(R_i). \qquad (5)\end{aligned}$$

In order to perform the portfolio optimization, we consider the case that an investor seeks to pursue a conservative portfolio management strategy such that he desires to minimize the portfolio variance subject to his desired return level and a zero value-at-risk level constraint. This is achievable when we set that the quadratic programming algorithm for the investor portfolio optimization to be

$$\min \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_i \sigma_j \text{ subject to } \hat{R}_p = \left(1 - \frac{\phi^{-1}(\beta)}{\phi^{-1}(\alpha)}\right)\sum_{i=1}^{N} w_i E(R_i), 1 = \sum_{i=1}^{N} w_i \text{ and } w_i \geq 0.$$

This quadratic programming algorithm seeks to minimize the quadratic objective function of the portfolio variance subject to the linear return, $\hat{R}_p$ constraints. For the estimation of the asset returns, an ensemble learning technique is adopted by using an artificial neural network with the input predictors being the estimates of OU and GBM models. Using a linear activation function for the neural network, assuming 4 hidden layers, we have that:

$$y = \phi_0 + \sum_{i=1}^{4} \phi_i(\phi_{0,1} + \sum_{i=1}^{2} \phi_{i,1} E(S_{t,i})),$$

where ϕ_0 denotes the intercept of the output neuron, $\phi_{0,1}$ denotes the hidden layer node output, $\phi_{i,1}$ denotes the weights corresponding to the hidden layer neuron nodes, y represents the output of the artificial neural network, and $S_{t,1}$ and $S_{t,2}$ represent the input node estimates of the OU and GBM models at observation time t. These estimates are respectively modelled as follows in (6) and (7):

$$\begin{aligned} S_{t,1} &= S_{0,1}\mathrm{e}^{-\theta t} + \mu(1 - \mathrm{e}^{-\theta t}) + \sigma \int_0^t \mathrm{e}^{-\theta t} dW_{t,1}, \qquad (6) \\ S_{t,2} &= S_{0,2}\mathrm{e}^{\left(\alpha - \frac{\gamma^2}{2}\right)t + \gamma dW_{t,2}} \qquad (7)\end{aligned}$$

In the OU model, the parameters θ, μ and σ represent the mean-reversion speed, long-run mean and volatility rate of the model. The parameters α and γ represent the drift and volatility of the geometric Brownian motion model. $dW_{t,1}$ and $dW_{t,2}$ are the standard Wiener processes for the OU and GBM models respectively. The means of these models $E(S_{t,1})$ and $E(S_{t,2})$ are used as the model estimates in the artificial neural network and are obtained by

$$\begin{aligned} E(S_{t,1}) &= S_{0,1}\mathrm{e}^{-\theta t} + \mu(1 - e^{-\theta t}), \qquad (8) \\ E(S_{t,2}) &= S_{0,2}e^{\alpha t}. \qquad (9)\end{aligned}$$

3 Implementation and testing

This section illustrates the implementation, testing and results from the hybrid predictive portfolio allocation prototype. The prototype is developed by using an artificial neural network with 4 hidden layers and a probability based quadratic optimization formula to obtain portfolio allocation weights. The inputs into the ensemble artificial neural network are the prices estimated by the OU and GBM models and are used to train the artificial neural network. The data used consists of 5 assets for a period of 5 years to train the artificial neural network. The prototype also uses the following data inputs, namely, the level target return, $\hat{R}_p$, the certainty probability of target return, β, the probability of zero loss exceedance, α, and the holding duration, h, as described in Section 2 (Code available on Github: [14]). The implementation of the artificial neural network, the portfolio optimization algorithm and the prototype software integrations are discussed in the subsequent subsections.

3.1 Artificial neural network and probability quadratic optimization algorithm

The prototype implements the artificial neural network algorithm in R and interfaces it into Python using the rpy2 interface package. The artificial neural network uses 4 hidden layers. The output layer consists of the prediction outcomes of the stock prices of the 5 assets. The input layer consists of estimates of the stock prices by OU and GBM models. The estimate of the node values of the artificial neural networks are obtained and are shown in the plots of Figure 3. The output results of the artificial neural network are subsequently used to obtain the portfolio allocations by the probability based quadratic optimization algorithm. This algorithm provides the various allocation weights for the assets given the input parameters of the level target return, the certainty probability of target return, the probability of zero loss exceedance and the duration. This algorithm also provides the portfolio risk (standard deviation) that will be attained by the portfolio for the given duration with the obtained allocation weights. The results of this algorithm using various inputs are shown in Table 1.

3.2 Software integrations

The prototype architecture integrates R-libraries and functionality within Python environment. The prototype database used to save predictions of stock prices and portfolio allocations is SQLite. The prototype also integrates with Microsoft Excel by allowing data in CSV format to be uploaded and used in the prediction and allocation of portfolio weights. Also, stored data of predictions and weights can be extracted and stored separately as a CSV file. Once the minimum hardware and software requirements of the computing device are met, the software prototype is installed in executable (.exe) format and the respective files extracted into the Windows OS filesystem.

4 Results and discussion

In this section, we present the results of the implementation of the robo-financial advisor prototype which applied ensemble artificial neural networks to forecast asset returns and consequently applied a probability based portfolio optimization algorithm to obtain the optimal portfolio allocation strategy for an investor. The results of the fitted ensemble artificial neural network algorithm, whose inputs are the OU and GBM models that are specified in Section 2, are presented in Figure 3. In these figures, the hidden layer consists of four nodes which are fully connected to every node in the input layer and the output layer. Each node performs a weighted sum of the inputs it receives from the input nodes and applies the activation function to produce the predicted prices. The results of implementation of the prediction algorithm and the portfolio allocation algorithm in the prototype using 5 assets and having varying inputs of portfolio target return, portfolio zero loss exceedance (value at risk level), certainty target return probability level, and the different durations are summarized in the Table 1.

As shown in Figure 2, the plots of the predicted stock prices using the ensemble artificial neural network are provided for the different selected assets, namely, the SPDR Dow Jones REIT ETF (RWR), iShares Russell 2000 ETF (IWM), S&P 500 (GSPC), iShares MSCI EAFE ETF (EFA) and iShares MSCI Frontier 100 ETF (FM). The predicted prices are compared with the historical estimates of a 60-day back-testing window. Generally, the artificial neural network algorithm has a strong ability to forecast stock prices accurately over the 60 day back-test period. These results are presented in the plots in Figure 2 below.

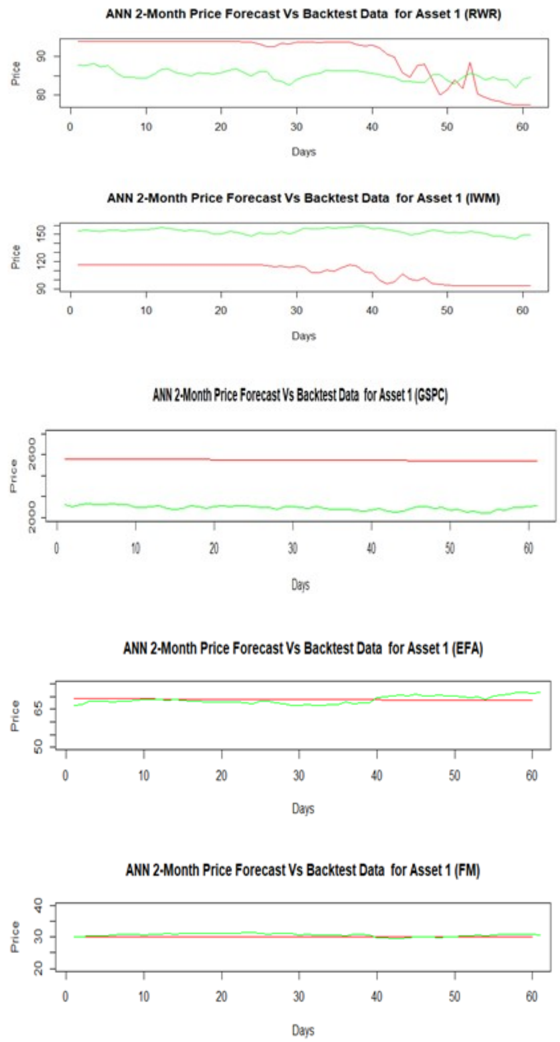

Figure 2: *Time series plot of predicted stock prices (historical prices in green, artificial neural network 60-day forecasts in red).*

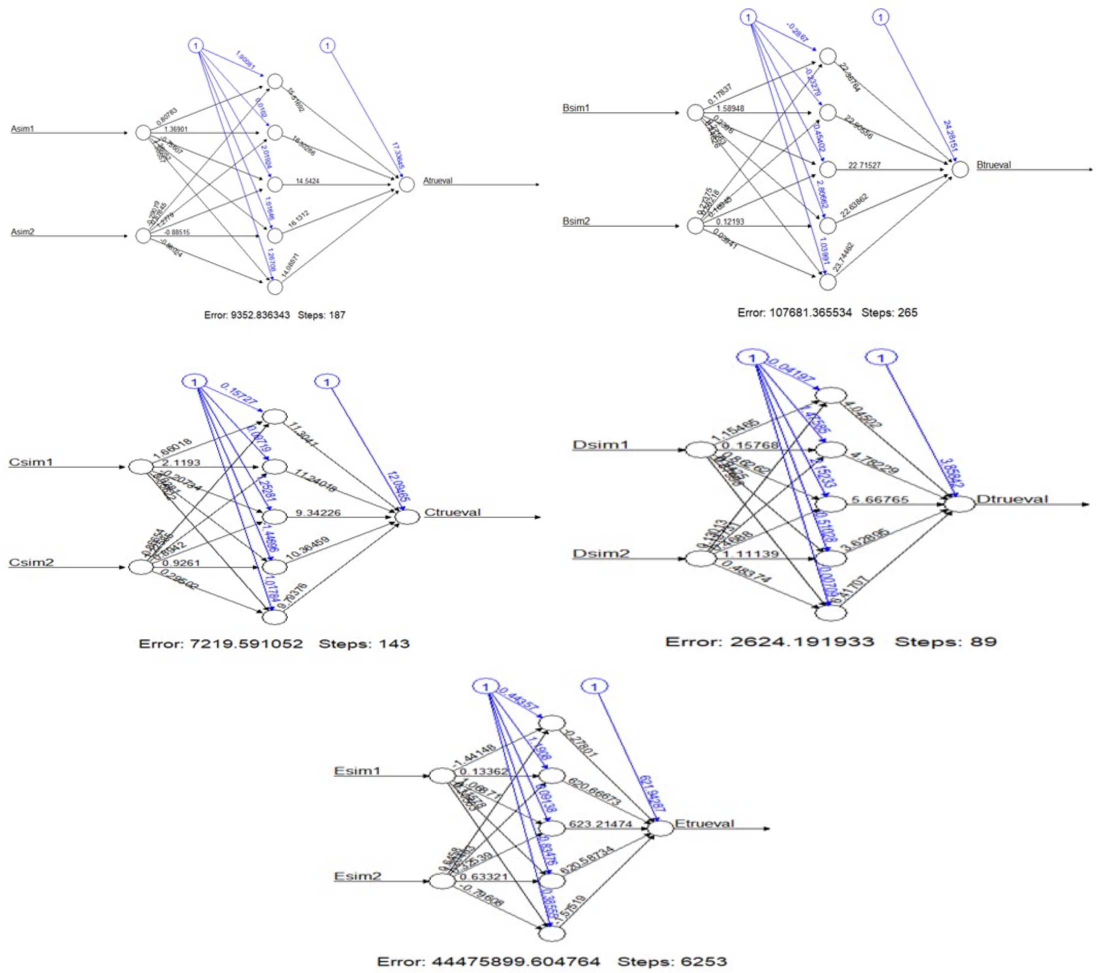

Figure 3: *Artificial neural network from training models (OU, GBM) using the training data set of stock prices of RWR, IWM, EFA, FM and GSPC which represent the OU and GBM estimates of the 5 assets respectively.*

The quadratic programming algorithm using the neural network forecasts could not converge to a local optimum at significantly high expected return levels to yield portfolio weights. The algorithm however converged to a local optimum at very low expected return levels to yield the portfolio weights. The returns produced by the neural network forecasting algorithm were very low, close to zero, consistently as shown in Figure 2, which compares the historical prices and artificial neural network forecasts. The OU and GBM estimation models produce slightly larger returns with larger standard deviations. Thus, the probability-based quadratic programming algorithm performed well when forecast returns have a high standard deviation or when the target return is low. A validation test with historical data and OU forecasts corroborated this, as it could converge when the target return level is generally high, particularly factors of 2, 3 times greater than target return desired in the analysis. These results of the allocation weights obtained by the quadratic programming algorithm using ANN predictions are shown in Table 1.

Generally, the neural network algorithm under-predicted the asset prices as shown in Figure 2 depicting the 2-month neural network forecasts against the historical test data. It accurately predicted the level of two of the assets (EFA and FM) and also correctly predicted the trend of two of the assets (RWR and IWM). This neural network algorithm was found to be much more suitable for short-term (30-day) forecasts rather than longer term (1 year) forecasts, as in the latter, it yielded very low, close to zero, standard deviations and consequently very low returns. The OU and GBM model forecasts had higher standard deviations than the artificial neural network forecasts.

Asset 4 (FM), in all algorithm test cases, received the highest allocation weight, except when the target return level was doubled and the probability levels of the target return and value-at-risk held the same as summarized in Table I below. The lowest variance portfolio was a medium-term portfolio with variance of 0.334%. However, its expected target return was -0.045% at 95% target return confidence. The highest variance portfolio was a short-term portfolio with variance of 1.542699%. Its expected target return 0.56% at 0.000001% target return confidence restriction. The highest allocation for an asset was to asset 1 (RWR) at a weight of 0.9873570 in the short-term duration.

Shorter time horizon investments were supported by the quadratic optimization algorithm since portfolio risk was lowest. The best diverse portfolio allocation was achieved in these shorter time horizons as well. Longer horizons allocations were biased towards assets with lower standard deviations, as in Asset 4 and Asset 5 allocation in Table 1. The lowest risk portfolios were the ones with a lower certainty probability of target return and vice versa. Also, the portfolio allocations were sensitive to the investment horizon. Shorter investment horizons were biased towards assets with the lower standard deviations. As shown in Table 1, Asset 1 received the highest allocation of 98% in the short-term. The medium-term investment yielded a well-diversified allocation. Asset 4 (EFA) consistently received a positive weighting in all algorithm allocations. Generally, for long-term investment, one would have to hold a slightly higher risk portfolio as compared to short-term investment.

5 Conclusion

The aim of this paper was to develop a robo-financial advisor prototype which adopts a hybrid programming architecture and uses artificial neural networks to forecast asset returns. It also sought to provide an optimal portfolio allocation strategy through a novel probability/value-at-risk optimization approach that minimizes portfolio risk within the prescribed investment horizons, given an expected portfolio return, budget constraints, certainty probability of target return, and value-at-risk level. Firstly, the factors relating to portfolio selection were investigated. This was achieved by conducting a comprehensive literature review to establish the factors that inform portfolio allocation. It is found that the portfolio returns, duration and the portfolio standard deviations are the main factors that would be considered in determining the portfolio allocation. This summary is presented in Section 1. The techniques applied in existing portfolio allocation platforms are investigated and it was found out that the Markowitz mean-variance approach was the most commonly used in existing robo-advisors for portfolio allocation. Secondly, an ensemble algorithm that could combine the prediction of stock prices for an investment

Probability of Target Return		**0.9**	**0.6**	**0.3**	**0.0000001**
Value at risk level		0.95	0.95	0.95	0.95
Portfolio Target Return Level		-0.001178916	-0.0007646989	-0.0004513797	0.002794288
Short-Term (0.5 months), 1*E(R)	[RWR] [IWM] [GSPC] [EFA] [FM]	NA, NA	[0.0000000] [0.0000000] [0.0000000] [0.3432342] [0.6567658] StdDev 0.006790228	[0.0000000] [0.0000000] [0.0000000] [0.3432342] [0.6567658] StdDev 0.004008075	NA,NA
2*E(R)	[RWR] [IWM] [GSPC] [EFA] [FM]	[0.9873570] [0.0000000] [0.0000000] [0.01264298] [0.0000000] StdDev 0.01452295	[0.9873570] [0.0000000] [0.0000000] [0.01264298] [0.0000000] StdDev 0.009420255	[0.9873570] [0.0000000] [0.0000000] [0.01264298] [0.0000000] StdDev 0.005560504	[0.9873570] [0.0000000] [0.0000000] [0.01264298] [0.0000000] StdDev 0.01542699
Medium-Term (2 months), 1*E(R)	[RWR] [IWM] [GSPC] [EFA] [FM]	[0.1122403] [0.0000000] [0.2188401] [0.5400799] [0.1288396] StdDev 0.008824622	[0.1122403] [0.0000000] [0.2188401] [0.5400799] [0.1288396] StdDev 0.005724055	[0.1122403] [0.0000000] [0.2188401] [0.5400799] [0.1288396] StdDev 0.003378745	[0.1122403] [0.0000000] [0.2188401] [0.5400799] [0.1288396] StdDev 0.009373945
2*E(R)	2	NA,NA	NA, NA	NA, NA	NA, NA
Long-Term (6 months), 1*E(R)	[RWR] [IWM] [GSPC] [EFA] [FM]	[0.0000000] [0.0000000] [0.0000000] [0.5222647] [0.4777353] StdDev 0.01039993	[0.0000000] [0.0000000] [0.0000000] [0.5222647] [0.4777353] StdDev 0.006745874	[0.0000000] [0.0000000] [0.0000000] [0.5222647] [0.4777353] StdDev 0.003981895	[0.0000000] [0.0000000] [0.0000000] [0.5222647] [0.4777353] StdDev 0.01104732
2*E(R)	2	NA, NA	NA, NA	NA, NA	NA. NA

Table 1: *Results of stock allocation (Asset 1 - RWR, Asset 2 - IWM, Asset 3 - EFA, Asset 4 - FM, Asset 5 - GSPC).*

duration using neural networks to achieve accurate predictions, where the inputs to the neural network model are stock price estimates from OU and GBM, is suggested in Section 2.

This ensemble artificial neural network prediction results were then fed into a probability-based quadratic optimization algorithm that takes into account the duration of investment, the certainty probability of target return, the probability of achieving a loss more than zero (value-at-risk) and the individual stocks standard deviation in order to obtain portfolio allocation weights that will yield a portfolio with low risk. The results of these algorithms are presented in Section 4. The developed prototype performed very well in estimating the portfolio allocations and performing predictions.

The results showed that the ensemble neural network algorithm correctly predicts the level of 2 of 5 assets and the trends of the remaining 3 assets; however, it yields low standard deviations and low returns compared to the OU and GBM estimates for short horizons. The neural network algorithm relies on these estimates as the training inputs, therefore recurrently optimizing the weights on the nodes. This shows that the ensemble artificial neural network-based approach to forecasting could be a reliable technique to forecasting, albeit at a short duration of about 30 days, since at longer horizons and a greater number of hidden nodes, the algorithm seems to be static even with heavy shocks to the underlying inputs.

The quadratic optimization algorithm is found to be sensitive to the level of the target return selected. If the target returns are very high or if individual stock returns are consistently zero, then the algorithm does not converge to local optimum to yield optimal portfolio weights. Setting of the target return within twice the actual historical portfolio return, but not higher, was found to lead to convergence and give the portfolio allocation weights. This algorithm supports investment in shorter time horizons since portfolio risk was lowest in the shorter time horizon. The best well-diversified portfolio allocation was achieved in these shorter time horizons. Longer horizons allocations were biased towards assets with lower standard deviations. The portfolios with the lowest risk portfolios were the ones with a lower certainty probability of target return, while the portfolios with the highest risk were the ones with a higher certainty probability of target return.

Finally, the results on the functionality and design of the prototype are also analyzed and presented. They show that a hybrid programming paradigm is an effective approach to leverage on strengths, speed and functionality of different programming languages; an elixir for multifaceted dissociable programming problems that can be implemented in compatible programming languages such as R and Python. The developed prototype was able to use, within Python, advanced R-based quadratic optimization, calibration and neural network libraries for portfolio analysis that are not available for Python, significantly improving application development speed and ease. Future investigative works could consider non-linearity models and explanatory inputs such as macro-variables to forecast returns and variances for longer durations. A hybrid approach on GPU to directly offload intensive algorithms for speed acceleration could also be considered as a possibility to improve the performance of the prototype.

References

[1] Ardia D, Mullen K, Peterson B, Ulrich J, & Boudt K, 2022, *DEoptim: Global optimization by differential evolution.*, R package version 4.1.0. URL: https://CRAN.R-project.org/package=DEoptim.

[2] Beketov M, Lehman K & Wittke M, 2018, *Robo advisors: Quantitative methods inside the robots*, Journal of Asset Management, **19(5)**, pp. 363-370.

[3] Business Insider, 1999, *Robo-Advisors: online Financial Advisors that Fit in Your Pocket.*, Business Insider.

[4] Deloitte, 2016a, *The expansion of robo-advisors in wealth management*, London: Deloitte White Paper.

[5] Deloitte, 2016a, *Robo-advisory in wealth management*, London: Deloitte White Paper.

[6] Djehiche Y, 2018, *A neural networks approach to portfolio choice*, KTH Royal Institute Working Paper.

[7] Freitas FD, Souza AF & Gomes FJ, 2009, *Portfolio selection with predicted returns using neural networks.*, Federal University of Espirito Santo Working Paper.

[8] Fritsch S, Guenther F, Wright M, Suling M & Mueller S, 1999, *neuralnet: Training of neural networks.*, R package version 4.1.2. URL: https://CRAN.R-project.org/package=neuralnet.

[9] Gautier L, 2020, *rpy2: Python interface to the R language.*, R package version 4.0. URL: https://rpy2.github.io/doc/latest/html/overview.html

[10] Grothey G, 2009, *Parallel computing: Financial applications in parallel portfolio optimization*, *Grouping multidimensional data*, Springer, Berlin.

[11] Hatemi J, & El-Khatib Y, 1999, *Portfolio Selection: An Alternative Approach*, Economics Letters, **135**, pp. 424–427.

[12] Kotu V & Deshpande B, 2018, *Data Science: Concepts and Practice (2nd Edition). Elsevier: Morgan Kaufmann Publishers.*

[13] Lam JW & Swensen A, 2016, *Robo-advisors: A portfolio management perspective*, Yale University Thesis.

[14] Muganda B & Kasamani B, 2023, *rpy2: Hybrid portfolio optimization with R and Python.*, GitHub Repository. URL: https://github.com/bwmuganda/hybridprogramming-rpy2

[15] Niiio Finance Group, 2018, *Robo-Advisory in the Securities Market: A Guide to Opportunity Assessment and Implementation*, Niiio Square Group.

[16] Peterson B, Carl P, Boudt K, Bennett R, Varon H, Yollin G & Martin R, 2004, *PortfolioAnalytics: Portfolio analysis, including numerical methods for optimization of portfolios.*, R package version 4.0. URL: https://CRAN.R-project.org/package=PortfolioAnalytics.

[17] Remillard B, 2013, *SMFI5: R functions for use with the SMFI5 software*, R package version 4.1.0. URL: https://CRAN.R-project.org/package=SMFI5

[18] Renaux MC, Rudnicki M, Beniere A, & Lorain F, 2019, *How will Robo-advisors Reshape Asset Management?*, Initio Square Group.

[19] Statista, 2022, *Robo advisors*, Statista Online Portal: Worldwide Market Study.