2020

A hybrid predictive prototype for portfolio selection using probability based programming and neural networks.

Muganda, Brian Wesley
Faculty of Information Technology
Strathmore University

A Hybrid Predictive Prototype for Portfolio Selection using Probability-based Quadratic Programming and Neural Networks

By

Brian Wesley Muganda

072559

A Dissertation Submitted to the Faculty of Information in partial fulfillment of the requirements for the award of Master of Science in Computing and Information Systems

MSc. Computing and Information Systems

Strathmore University

April 2020

Declaration and Approval

I Brian W. Muganda declare that this research has not been submitted to any other University for the award of a Master's Degree in Computing and Information Systems.

Student Name: BRIAN WESLEY MUGANDA

Sign: _____ Date: _____

Supervisor's Name: DR. BERNARD SHIBWABO

Sign: _____ Date: _____

Abstract

A portfolio is a collection of investments held by an investment company, hedge fund, financial institution or individual. This collection of investment features a combination of financial assets such as stocks, bonds or options. The designing of a portfolio (fund allocation to each asset and selection of the assets) is done according to the investor's risk tolerance, investment time frame and investment objectives. Robo-advisors, which are automated algorithm-driven investment platforms that use quantitative algorithms to manage investors' portfolios, are at times used to perform portfolio allocation for investors having defined their risk preferences and investment time frames. However a majority of these robo-advisors still rely on classical mean variance allocation techniques of modern portfolio theory. This research therefore, developed a robo-financial advisor prototype on a hybrid programming architecture by using artificial neural networks to predict portfolio returns and variances based on underlying multi-asset Uhlenbeck process (OU) and geometric brownian motion (GBM) processes' estimates. These results were subsequently used in the probability-based quadratic optimization algorithm to provide an optimal portfolio allocation strategy. This probability-based quadratic programming approach is novel and is based on return certainty probability and value-at-risk constraints acting as proxies for investor's risk tolerance. The results showed that neural network algorithm performed averagely well forecasting being able to predict the correct level of 2 of 5 assets and to predict correctly the trends of the remaining 3 assets, it however yielded low standard deviations compared to the OU and GBM models. The quadratic optimization algorithm supported investment in shorter time horizons since portfolio risk was lowest. Diversified allocation was achieved in the shorter time horizons. Longer horizons allocations were biased towards asset with lower standard deviations. Lowest risk portfolio was the ones with a lower certainty probability of target return and vice versa. Also, it shows 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.

Keywords: Artificial Neural Networks, Prediction, VAR –Constrained Optimization, Certainty Constraint, Portfolio Selection

Table of Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

ANN	Artificial Neural Networks
AR	Autoregressive Model
ARIMA	Autoregressive Integrated Moving Average Model
CVaR	Conditional Value at Risk
ETF	Exchange Traded Funds
FINTECH	Financial Technology Companies
FNN	Feed-forward Neural Networks
GBM	Geometric Brownian Motion process
MA	Moving Average Model
MPT	Modern Portfolio Theory
MVO	Mean Variance Optimization
OU	Ornstein–Uhlenbeck mean reverting process
RA	Robo-advisor
RNN	Recurrent Neural Networks
SDE	Stochastic Differential Equation
VaR	Value at Risk

Definition of Terms

Artificial neural network	A computational model that consists of several processing elements that receive inputs and deliver outputs based on their predefined activation functions.
Exchange Traded Fund	A type of security that involves a collection of securities—such as stocks, bond or commodities—that often tracks an underlying index, although could be in any number of industry sectors or use other various strategies.
FinTech	Companies or services that use innovation and technology to provide financial services to businesses or consumers.
GBM Process	A continuous-time stochastic process in which the logarithm of the randomly varying quantity follows a Brownian motion with drift.
Modern Portfolio Theory	A portfolio analysis framework for assembling a portfolio of assets such that the expected return is maximized for a given level of risk or the level of risk in minimized subject to an expected level of return.
OU Process	A continuous stationary Gaussian mean-reverting process which tends to drift toward it long-term mean.
Portfolio	A collection of investments held by an investment company, hedge fund, financial institution or individual. This collection of investment features a combination of financial assets such as stocks, bonds or options.
Robo-advisor	An automated financial adviser that provides financial advice or investment management online with moderate to minimal human intervention. It provides digital financial investment management advice based on mathematical rules or algorithms.
Value-at-Risk (VaR)	A measure of the risk of loss for investments. It estimates how much a set of investments might lose with a given probability, given normal market conditions and a set time period such as a day.

Chapter 1 : Introduction

1.1 Background

A portfolio is a collection of investments held by an investment company, hedge fund, financial institution or individual. This collection of investment features a combination of financial assets such as stocks, bonds or options. The holder of the portfolio can have the portfolio managed by financial professionals or other financial institutions on his behalf for a management fee. The designing of a portfolio (fund allocation to each asset and selection of the assets) is done according to the investor's risk tolerance, investment time frame and investment objectives. Investors have varying risk preferences, investment time frames and investment objectives thus a variety of methods may be applied to design this portfolio to have proper fund allocation to each asset and select assets that meet the investor's objectives (Hatemi & El-Khatib, 2015; Lam & Swensen, 2016; Statista, 2018).

A robo-advisor can be defined as an automated investment platform that uses quantitative algorithms to manage investors' portfolios and is accessible to clients. They provide automated, algorithm-driven financial planning services with little to no human supervision. Robo-advisor digitally onboards clients while collecting information about them, their financial situation and future investment objectives. After having assessed the client's risk profile, it uses this data, combined with market data, to offer advice and/or automatically allocate and rebalance clients' portfolios. A Robo-advisor can also be used as a support to portfolio managers in their decision-making process, providing them with smart and automated recommendations, alerts and dashboards (Renaux et al., 2019; Lam & Swensen, 2016).

The clients' portfolio is derived by using a risk profiling process, and is usually mapped by the robo-advisor via a cost-effective Exchange Traded Fund (ETF) portfolio. This approach contrasts considerably with the banks' relatively opaque, expensive and non-digital securities products. Due to the high degree of automation and the mapping of portfolios by means of cost-efficient Exchange Traded Funds (ETFs), robo-advisors can offer their services within considerably better developed conditions than traditional asset management companies (NIIO, 2018).

The first- and second-generation robo-advisory systems (RAs) comprise online questionnaires and proposals, thereby providing a combination of advice and online access to traditional ‘‘manual’’ asset management services. In contrast, the third and fourth generations of RAs use quantitative methods and algorithms to construct and rebalance the portfolios, thereby performing truly automated portfolio management. The differences between these generations are only the level of automation and methodological advances. Hence, the third- and fourth-generation RAs are the systems that cover the entire investment and portfolio management process, starting from the selection of the asset portfolio and finishing with periodic portfolio rebalancing and appropriate performance reporting (Deloitte, 2016b).

The rise of robo-advisory platforms for wealth and asset management can be attributed to a new generation of clients who are receptive to digital technologies, are well educated, prefer to have active and ongoing control over their investments, and rely on the information from multiple sources rather than individual financial advisors. This demographic change is supplemented by changes in older generations who are also becoming increasingly technologically receptive, thereby expecting and demanding digital investments and advisory services (Deloitte, 2016a).

Robo-advisory services can be offered at much lower costs in comparison with traditional human advisors. They can also achieve approximately the same returns on investments as traditional advisors. Users of robo-advisors such as clients or fund managers have access to various options to control, customize, and construct investment portfolios from multiple devices such smartphones or laptops. These services also have transparent workflow and monitoring systems, require a low or even no minimum investment, and utilize advanced quantitative methods of portfolio management and optimization (Beketov, Lehmann & Wittke, 2018).

A forecast by Statista (2007) predicts that RAs will globally manage between $0.8 and $8.1 trillion by 2020, which is 1–10% of the total global assets under management. 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. The main reasons for the current success of robo-advisory systems can be attributed to this new generation of clients and the advantages of these systems over traditional financial advisors.

Robo-advisors are at the crossing of different strong trends: lower management fees, digital distribution and machine-enhanced decision making. This, coupled with, allowing for the no strict requirement on the minimum investment in robo-advised funds; has enabled RAs to present a seductive alternative investment solution for many people who would not be able to qualify for the high-end products available in traditional private banking or asset management (Renaux et al., 2019).

The formula for success concerning robo-advisors focuses on transparent, cost-effective - and above all - digital product (technological) offerings. In order to compete effectively with FinTechs, banks increasingly pursue cooperation agreements with FinTechs, and offer their own customers robo-advisors in the form of white-label solutions. The benefit of this approach is that white-label solutions can be implemented much faster than in-house developments as they are designed upon the existing expertise of FinTechs. Additionally, many FinTechs have also expanded their business model to include cooperation agreements with banks, with some focusing exclusively on RAs services. Consequently, the probability of a sustainable product success rate is low, given this competition and the fact that the banks' RAs are more expensive to use. 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 (NIIO, 2018; Statista, 2018; Business Insider, 2017).

1.2 Problem Statement

Investors are usually limited by cognitive and emotional biases in their investment decision making and with that end up making poor portfolio investment decisions. That then is why they need to rely on more and more sophisticated tools to enhance their decision making. Robo-advisors can assist in overcoming these biases and remain more objective, since, if they are specified and developed correctly, they are not subject to these biases (Renaux et al., 2019).

Modern Portfolio Theory remains the main framework used in these robo-advisor systems as noted by Beketov, Lehmann and Wittke (2018). They found that the current trend is to attempt to improve and augment this framework rather than applying and developing entirely new approaches. There exists consequently a clear gap between the predominant methods applied in

robo-advisors and new methodological developments. The marketing, cost cutting and performance benefits of adopting such a robo-advisor incorporating new methodological approaches is profound. The clients investing using these robo-advisor systems are typically receptive to technology and would demand both sophisticated methods and digital services. They would expect that their investments will be managed with advanced, scientifically justified, modern, and well-implemented methods and technology.

Therefore, the jointure of advanced quantitative methods and efficiency requirement for such robo-advisory systems for widened portfolios of about 20 or more assets, demands of course the appropriate computation power to progressively be able to produce results for series of complex computational methods. This requirement would be achieved by adoption of much complex approach to portfolio allocation involving multi-asset OU and GBM models incorporation, training of a neural network algorithm and the quadratic optimization techniques to obtain optimal allocation strategies. The value proposition of efficiency of such a prototype would be achieved by using a hybrid programming architecture. A low-latency and traditional optimization methods on a robo-advisor would not provide value of use for clients that are interested in high quality scientific and efficient robo-advisory services (Verelst, 2019).

1.3 Aim

The aim of this research is to develop a robo-financial advisor prototype which adopts a hybrid programming architecture and uses machine learning techniques to provide an optimal portfolio allocation strategy that maximizes the returns of investors for the prescribed investment horizons and budget constraints. The prototype should utilize a weighting of prediction algorithms by using artificial neural networks to predict the expected price of assets in the portfolio. The underlyings in the prediction algorithm should be multi-asset OU and GBM of models. It should then apply probability-based quadratic optimization algorithm to provide the optimal portfolio allocation strategy that maximizes on investors returns in the given investment horizon.

1.4 Research Objectives

i) To analyze the factors relating to portfolio allocation using robo-advisors
ii) To investigate the existing modelling techniques used for returns prediction and portfolio allocation

iii) To develop a hybrid predictive portfolio allocation prototype that use artificial neural networks to forecast asset returns and a probability-based quadratic model for optimization

iv) To test the developed prototype

1.5 Research Questions

i) What are the factors that inform portfolio allocation decisions and use of robo-advisors?

ii) Which stock return prediction and portfolio allocation techniques are used in existing robo-advisors?

iii) How can a hybrid predictive prototype that uses artificial neural networks for return prediction and probability-based quadratic optimization to determine optimal portfolio allocations be developed?

iv) How effective is the developed prototype?

1.6 Justification

Robo-advisors typically provide services that go beyond simple advisory services to encompass comprehensive portfolio management services that allow individuals to plan and delegate their investment decisions. Mean-variance is the predominant model used in robo-advisors globally. Having robo-advisors that can provide accurate and optimal portfolio return maximizing portfolios, could help in cost cutting and marketing of these platforms by financial services companies (Business Insider, 2017). Clients can have access to various options to control, customize, and construct investment portfolios from multiple devices such smartphones or laptops at much lower costs. The robo-advisors can present a seductive alternative investment solution for many people who would not be able to qualify for the high-end products available in traditional private banking or asset management. Fund managers could also benefit machine-enhanced real-time decision making (Hatemi & El-Khatib, 2015; Lam & Swensen, 2016; Statista, 2018; Business Insider, 2017).

Mean-variance optimization approach to portfolio selection has some shortcomings such as assumption on normality, optimization solely based on first two moments of return distribution, does not consider risk tolerance measures or views or certainty on portfolio performance or the value-at-risk portfolio performance reporting regulations. It also uses historical returns and thus

invariably prone to estimation errors which lead to selection of inefficient portfolios (Hatemi & El-Khatib, 2015). Therefore it would be of interest in academia on how to efficiently predict portfolio returns using neural networks and use of a novel approach on allocation that considers the certainty of the returns / investor risk tolerance and value-at-risk required threshold in a parallel performance enhancing computing environment.

1.7 Scope and Limitation

The proposed decentralized machine learning prototype is tested using a preset selection of 5 assets, whose allocation would then be determined using probability-based quadratic optimization and artificial neural network prediction.

Chapter 2 : Literature Review

2.1 Overview

This section of this proposal reviews existing related literature and previous research and studies on asset management robo-advisors, probability-based asset allocation and use of prediction models in portfolio allocation.

2.2 Theoretical Review

2.2.1 Portfolio Selection Theory

Mean-variance optimization, introduced by Harry Markowitz in 1952 was the first mathematical formalization of the idea of diversification of investments. The framework considers a set of risky assets and calculated portfolios for which the expected return is maximized for a given level of portfolio risk, where risk is measured as variance. In an alternative formulation, optimization involves minimizing portfolio risk for a given level of expected return. These optimized portfolios compose the "efficient frontier," which is a band of portfolios that dominate all other feasible portfolios in terms of their risk-return tradeoff. The primary benefit realized by the utilization of mean-variance optimization is portfolio diversification (Lam & Swensen, 2016).

The mean-variance optimization approach however takes into account the overall risk of the securities without separating out their systematic and idiosyncratic components. It also assumes that asset class returns are normally distributed, however actual returns possess significant non-normal characteristics. This normality assumption inadequately accounts for the possibility of extreme market movements; also with normality assumption which uses a symmetrical variance measure that does not distinguish between upside and downside moves; investment returns with positive skew will appear riskier than they really are, leading to under-allocation of the asset class; similarly, returns with negative skew will appear less risky than they really are, leading to over-allocation of the asset class.

Estimation error invariably leads to inefficient portfolios, which is evidenced by actual frontier, which is computed using the true expected returns but the weights of the portfolios from the *estimated frontier,* always lies below the *true efficient frontier.* Michaud (1989) notes that the mean-variance optimizers are, in a fundamental sense, 'estimation-error maximizers.' The risk

and return estimates are inevitably subject to estimation error. Mean-variance optimization significantly overweights (underweights) those securities that have large (small) estimated returns, negative (positive) correlations and small (large) variances.

As noted by Chopra (1993) assumptions about expected returns exert the largest effect in determining portfolio weights, while variances and covariances exert secondary and tertiary effects, respectively. Calculating the average portfolio turnover resulting from switching from a base portfolio to one based on error-tainted inputs. Chopra (1993) shows that for an investor with a moderate risk tolerance, the average turnover due to estimation errors in means is two to four times the average turnover from estimation errors in variances and about five to thirteen times the average turnover from errors in covariances. A separate study by Chopra and Ziemba (1993) corroborates these results. By examining the cash equivalent loss from optimizing portfolios based on estimated, rather than true, input parameters, they showed that for an investor with a moderate risk tolerance, errors in means are eleven times as damaging as errors in variances.

Errors in variances are twice as damaging as errors in covariances. Moreover, they found that the relative importance of errors in means, variances, and covariances depends upon the risk tolerance of the investor. Since an investor with higher risk tolerance focuses on raising the expected return of the portfolio while deemphasizing the variance, errors in expected return exert a larger effect on investment results. Conversely, the investor with a low risk tolerance focuses on reducing portfolio risk and hence is less affected by errors in means than the investor with higher risk tolerance (Chopra & Ziemba, 1993).

Lam and Swensen (2016) in their review 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 note that investors wishing to use the mean-variance optimization technique should consider further simulations that perturb the underlying parameters: mean, standard deviations, correlations, and VaR coefficients.

2.2.2 Asset Management Robo-advisors

Robo-advisors typically provide services that go beyond simple advisory services to encompass comprehensive portfolio management services that allow individuals to plan and delegate their investment decisions (Abraham, Schmukler & Tessada, 2019). According to this report by Abraham, Schmukler and Tessada (2019), in Europe, there are currently over 70 robo-advisors, with 5 of them managing more than €100 million and in the emerging economies, there has been an emergence of robo-advisors also. For example, the number of robo-advisors is growing fast in Asia, driven by an emerging middle class and high technological connectivity. Robo-advisors are already present in China, Hong Kong, India, Japan, Singapore, Thailand, and Vietnam, among other economies. Other emerging regions also have robo-advisors, but their presence is limited so far. There are for instance only 6 robo-advisors in Africa and Latin America altogether. However Robo-advisors are expected to continue expanding around the world in years to come. Some projections for example Business Insider (2017) have even forecasted that robo-advisors will manage around 10 percent of global investment assets by 2020.

Beketov, Lehmann and Wittke (2018) in their study analyzed 219 existing robo-advisors worldwide and found that Modern Portfolio Theory remains the main framework used in these robo-advisor systems. In their summary of the building blocks of robo-advisory systems, they noted that all systems use ETFs with minor exceptions being mutual/Actively Managed Funds, Sustainable Funds and Index Funds. In identification of risk profile of the clients, these systems studied used online questionnaires focused on identifying clients' risk tolerance as well as investment objectives and horizon. Typically, the questions were compiled to understand the objective risk tolerance through information on age, income, savings, previous investment experience, and investment goals. The asset allocation methodology adopted by 80% of these systems are Modern Portfolio Theory approach, about 8% supplemented with Black– Litterman, VaR and CVaR optimization techniques, 20% adopted monte carlo simulation approaches. Furthermore, these systems used event/threshold-based rebalancing based on the daily rebalancing check and some used cash-flow (re)investment for the rebalancing. Half of the systems studied provided control and monitoring possibilities through the website only. This half

also provided a smartphone app. Some systems sent the monthly statements and quarterly reports automatically by e-mail.

To help with investment decisions, these robo-advisors start by defining the investment strategy of each client based on their investment objectives and risk profile. It asks the potential clients about the purpose of their investment and the time horizon. It then offers investment strategies for a variety of these objectives such as retirement, saving for large expenditures, income stream generation, portfolio selection (Abraham, Schmukler & Tessada, 2019). Robo-advisors use automated algorithms to make recommendations on how to allocate funds across different types of assets. In most cases, these algorithms are based on modern portfolio theory (Bjernes & Vukovic, 2017).

In addition to recommending an initial allocation of funds, these algorithms can be designed to continuously monitor portfolios and detect deviations from the targeted risk. Whenever deviations are identified, the portfolio is automatically rebalanced. For example, the value of equity might increase faster than the value of bonds over time, increasing the share of the portfolio invested in equity. A higher share of equity increases risks, thus the portfolio might be rebalanced by selling equity. The portfolio can be automatically rebalanced to reduce risks as time goes by and as the investment horizon approaches and the fund therefore needed. This portfolio rebalancing can also be done when the investor changes their risk tolerance or investment objectives (Abraham, Schmukler & Tessada, 2019).

2.3 Empirical Review

2.3.1 Asset Prediction Algorithms for Portfolio Selection

Freitas, Souza and Gomes (2009) study portfolio selection under the Markowitz Model using expected returns predict using neural networks. Their model is a modified version of the Markowitz Model that uses time series prediction instead of first order statistical measurements. They used a neural network predictor to obtain an approximation of future prices and these values were used to estimate of future returns, which they used as expected returns on the Markowitz's Model. They then compared results from their model with the Markowitz model using real data and found that their model delivered a return 12.39% higher than the mean-variance model, they used 66 stocks from Brazilian BOVESPA stock exchange. Their results

also show that the neural network predictor, which uses and AR (4) model as the training model, can produce better estimates for future returns than the historical time series mean value.

Djehiche (2018) in his study investigates a neural networks approach to portfolio choice, opines that if the relationship between the output and input variables is non-linear, the linear regression model may not be a suitable choice for prediction of the return. He proposes use of an Artificial Neural Network (ANN) is a non-linear statistical model. He considers two different ANN models, Feed-forward Neural Networks (FNN) and Recurrent Neural Networks (RNN), these models are trained to predict monthly returns on asset data consisting of macroeconomic data and market data. The data set they used in the study was that of 13 different assets. The explanatory variables of each asset consist of macroeconomic data such as inflation, money supply and current account, and market data such as foreign exchange, yield curves and volatilities. The response variable is the corresponding monthly return. The predicted returns are then used in a long-short portfolio strategy. They then compared prediction performance of the networks and their corresponding portfolios to a benchmark linear regression model. Metrics such as average hit-rate, mean squared prediction error, portfolio value and risk-adjusted returns were used to evaluate the model performances. The linear regression and the feed-forward model were found to yield good average hit-rates and mean squared-errors, but poor portfolio performances. The recurrent neural network models yielded worse average hit-rates and mean squared prediction errors, but had outstanding portfolio performances.

2.3.2 Portfolio Selection and Prediction Prototype Systems Architecture

Bekkerman, Bilenko and Landford (2011) in their study of scaling up machine learning through parallel and distributed approaches, note that recent years have brought dramatic progress in usability, cost effectiveness, and diversity of parallel computing platforms, with their popularity growing for a broad set of data analysis and machine learning tasks. They attribute this current rise in interest in scaling up machine learning applications to the evolution of hardware architectures and programming frameworks that make it easy to exploit the types of parallelism realizable in many learning algorithms. A number of platforms have made it convenient to implement concurrent processing of data instances or their features.

Bekkerman, Bilenko and Landford (2011) suggest a number of reasons why practitioners may choose to machine learning task using parallelization. These are when there are large number of data instances such that the potential training examples are extremely large; Model and algorithm complexity where a number of high-accuracy learning algorithms either rely on complex, nonlinear models, or employ computationally expensive subroutines; Inference time constraints where applications require predictions to be made in real time; Prediction cascades where applications require sequential, interdependent predictions that have highly complex joint output spaces; and Model selection and parameter sweeps where tuning hyper-parameters of learning algorithms and statistical significance evaluation will require multiple executions of learning and inference.

Grothey (2009) in his study 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. He stresses that parallel processing becomes mandatory when we have to deal with multi-period portfolio optimization problems which are treated as stochastic programming problems, meaning optimization is performed over a selection of future scenarios. The desirability then of having future scenarios match static and dynamic correlations between assets for all future time periods leads to an optimization problem of an enormous size. In his approach, he implemented the stochastic programming problem by use of decomposition methods and interior point methods, on parallel machines, PC clusters and grid environments.

Balasubramanian (2010) is his study of parallel programming, states that parallel programming is increasingly becoming important for researchers and developers of large-scale distributed and parallel applications in a number of domains, including financial risk assessment and modeling (e.g., Value-At-Risk and historical calculations), real-time decision-making based on algorithmic feedback (e.g. market making, electronic strategy arbitrage, and high-frequency trading), and processing, archiving, storing and searching content repositories for enterprise content management systems. He further states that financial applications involving massive simulations, are well suited for distribution and parallelization. The prohibitive effort that is needed to

parallelize these applications using traditional mechanisms has restricted the financial industry's movement in this direction however real-time performance has become an increasingly critical factor in making timely trading decisions and to assure fast run-times for real-life portfolios problems.

Financial applications involving risk estimation by use of monte carlo methods which rely on simulations based on hypothetical market behavior scenarios have proven quite useful in risk calculations in the financial services industry. The computational intensity of such methods, however, generally limits the frequency with which they can be used. These predictions must be calculated to generate a sufficiently large number of scenarios for reliable statistical analysis. The computational cost then, of such calculations is often the factor limiting their broader use in the financial industry. Therefore, there would be a significant benefit from boosting the performance of such computations by developing parallel financial applications Balasubramanian (2010).

Singh, Barford and Harris Jr (2016) studied the acceleration of the Critical Line Algorithm for portfolio optimization by using NVIDIA'S GPU and CUDA API for the intensive parts of the algorithm. The critical line algorithm derives from Markowitz's theory focuses on efficient portfolio that yields the maximum return for a minimum risk (or volatility). When several efficient portfolios are computed, they make up the Efficient Frontier. There can be no portfolios that exceed the frontier, the optimal portfolio (max return or min variance) will be on the frontier and all other portfolios will be inside the boundary of the curve. The portfolios on the Efficient Frontier will dominate the rest of the portfolios, this is the Markowitz Efficiency Frontier.

They found that the parallelization of the algorithm on the GPU led to a speed up which grew in relation to the number of assets. They we able to attain speed ups of an asset population of 501 by 8X. As the size of the asset population increases, the speed up they attained in relation to the sequential completion time grew drastically. The method in which the Critical Line Algorithm was optimized, it could handle 65535 assets on one GPU before it will require additional GPUs for the matrix reduction. The sequential implementation runs at a much faster time for a small number of assets, however once at around 500 assets, the GPU method yielded higher speed up. They endorsed the utilization of the power of GPUs for intensive computational problems in finance.

2.4 Conceptual Framework

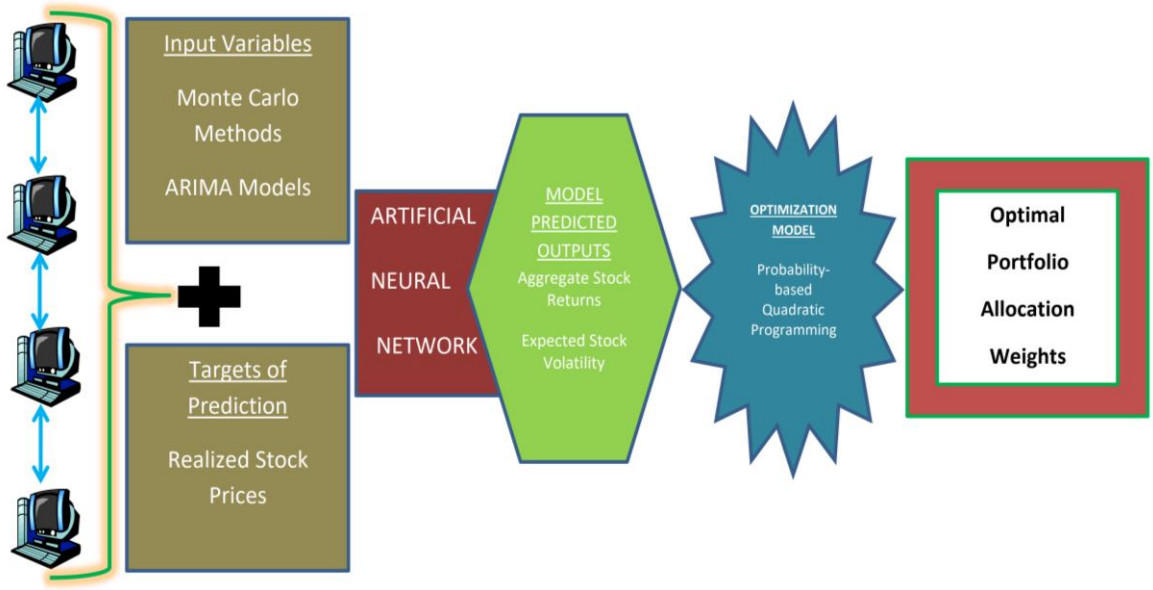

Figure 2.1: Conceptual Framework

The conceptual framework in Figure 2.1 describes the functioning on the hybrid predictive portfolio allocation prototype. Inputs of the artificial neural networks are OU and GBM prediction models with the target node being the realized stock variance. These inputs are used to train the neural network to conduct prediction of stock prices and stock returns. These achieved predictions of the neural network are then used in the probability-based quadratic optimization model so as to achieve the constrained optimal portfolio allocation with constraints being the probability adjusted target return, the desired value at risk probability and positive weights.

Chapter 3 : Research Methodology

3.1 Overview

This research aimed at developing a hybrid predictive prototype for portfolio selection. The methods used for conducting the research and its viability are described in this chapter. The target data population, the sample size to use in the research, data collection procedures, analysis methods of the results obtained and system development approach and tools are also discussed.

3.2 Research Design

This research adopted a quantitative design approach to determine the portfolio allocations by developing and testing a probability-based quadratic optimization model that adopted artificial neural networks with input models being of OU and GBM mean types. The prototype was tested by creating a portfolios of 5 assets each supposing that the users of this prototype were to invest in any of these portfolios. The prototype worked with a combination of 5 different assets in a portfolio.

3.3 Location of Study

This study was conducted in Strathmore University computer laboratories where the hybrid predictive portfolio allocation prototype was tested. The development environment is such that a multilanguage programming paradigm is adopted by interfacing Python and R to leverage of the speed and libraries of these programming languages respectively. The prototype works with data from any given financial market for a minimum of a year duration.

3.4 Target Population and Sample

The target data population for this study are the listed security prices at New York Stock Exchange (NYSE). The sample consisted of Exchange Traded Funds (ETFs) and Index Linked Funds for a period of 5 years. The selection of Exchange Traded Funds and Index Linked Funds was made since performance of these assets is representative of market volatility and is a preferred asset investment class for investors interested in portfolio diversification. A selection of 5 assets with the highest aggregated daily trading volumes.

3.5 Data Collection

Stock data was collected from Yahoo Finance of a portfolio of 5 Exchange Traded Funds (ETFs) and Index Funds. The data was analyzed to obtain return predictions and standard deviation estimates for the holding period of the investments. The 5-year time series data of ETFs and Index Funds is shown in Table 3.1 below. For larger portfolio allocation with more than 5 assets, a subset of 5 assets can be constructed recursively.

Table 3.1: Portfolio Data collected

Constructed Portfolio
SPDR Dow Jones REIT ETF (RWR)
iShares Russell 2000 ETF (IWM)
iShares MSCI EAFE ETF (EFA)
iShares MSCI Frontier 100 ETF (FM)
S&P 500 (GSPC)

3.6 Model Specification

In our approach to finding the optimal portfolio allocations, we move away from the utility maximization approach as has been used previously and suggest use of a risk measure probability based allocation approach whereby we aim at minimization of portfolio volatility subject to a least value-at-risk loss, a return maximum target and a budget constraint.

3.6.1 Probability-based Quadratic Programming

The return on an individual stock is defined as:

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

The portfolio return therefore is the sum of all the individual stock returns for a given holding period *h*,

$$R_p^h = \sum_{i=1}^{N} w_i \, R_i^h$$

The optimal portfolio model that the robo-advisor utilizes is defined as follows:

The target return portfolio $\hat{R}_p$ given some certainty (expectation) level of target return β is:

$$\hat{R}_p = E(R_p) + \phi^{-1}(\beta)\sqrt{var(R_p} \quad 3.1$$

Setting a condition that the value-at-risk should be at most zero, meaning that given only α probability level, the loss on the portfolio would exceed zero. We have then that:

$$0 = E(R_p) + \phi^{-1}(\alpha)\sqrt{var(R_p}$$

$$\sqrt{var(R_p)} = \frac{-E(R_p)}{\phi^{-1}(\alpha)} \quad 3.2$$

Thus from 3.2, the target return becomes:

$$\hat{R}_p = E(R_p) - E(R_p)\frac{\phi^{-1}(\beta)}{\phi^{-1}(\alpha)}$$

$$\hat{R}_p = \left(1 - \frac{\phi^{-1}(\beta)}{\phi^{-1}(\alpha)}\right) E(R_p)$$

Where the expected return of the portfolio

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

Therefore the target return with portfolio allocation weights w_i and individual expected asset return $E(R_i)$ is specified as:

$$\hat{R}_p = \left(1 - \frac{\phi^{-1}(\beta)}{\phi^{-1}(\alpha)}\right) \sum_{i=1}^{N} w_i\, E(R_i) \quad 3.3$$

Where the total allocation of weights sums to 1

$$1 = \sum_{i=1}^{N} w_i$$

We now define the probability-based quadratic programming algorithm given a conservative active portfolio management strategy where they portfolio manager or investor is interest in achieving minimum portfolio variance for a given probabilistic maximum target return and zero value at risk loss. The probability based quadratic programming algorithm is thus defined as:

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_i \sigma_j$$

$$subject\ to: \hat{R}_p = \left(1 - \frac{\phi^{-1}(\beta)}{\phi^{-1}(\alpha)}\right) \sum_{i=1}^{N} w_i\, E(R_i)$$

$$1 = \sum_{i=1}^{N} w_i \ \ and\ w_i \geq 0$$

3.6.2 Prediction of Expected Returns

Prigent (2007) opines that the Markowitz mean-variance analysis heavily relies on good prediction of the mean and variance-covariance matrix of the asset returns. Mean-variance optimal portfolios are significantly sensitive to parameter values. Using past data to predict future returns means that we assume stability of parameters through time. Generally, the variance-covariance matrix is more stable than the mean return. Therefore, we will here adopt weighted return prediction under artificial neural network model to achieve a stable mean return prediction; with underlying time series forecasting algorithms being OU and GBM models.

The elementary building blocks of the human nervous system are called neurons. Similarly, the building blocks of ANNs are called neurons (or nodes) and are based on Rosenblatt's single-layer perceptron. The neuron consists of a vector of multiple real-valued inputs $\boldsymbol{S} = (S_1, \ldots, S_n)$ and a single output S_t . The connection between a input value X_i and an output value S_t is indicated with a connection weight γ_i. The output is then obtained by computing the activation value U as the sum of S, with their respective weights in the vector $\gamma = (\gamma_1, \ldots, \gamma_n)$, and a bias term γ_0.

$$U = \gamma_0 + \sum_{i=1}^{N} \gamma_i S_i$$

Passing it through an activation function *f*, we have that:

$$S_t = f(U) = f(\gamma_0 + \sum_{i=1}^{N} \gamma_i S_i)$$

The values of S_i are obtained from the forecast of mean values of OU and GBM models to predict the stock price S_t. Selecting the identity function, *f(u)* = *u*, yields a multiple linear regression. Thus, linear regressions are a special case of neural networks. The non-linear activation functions are key parts of that give a non-linear ANN the ability to model non-linear functions. Figure 3.1 describes the architecture of an artificial neural network.

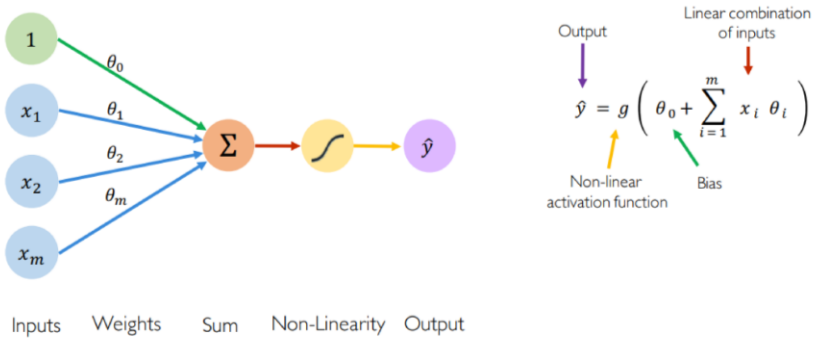

Figure 3.1: Architecture of an ANN

From the input layer nodes we thus have that:

$$y = f(w_0 + \sum_{i=1}^{2} w_i S_i)$$

Where $w_0$denotes the intercept (bias) of the output neuron, w_i are the weights of the covariates (input variables) and x_i are the covariates

Using a linear activation function f(x)=x, we have that :

$$y = w_0 + \sum_{i=1}^{2} w_i s_i$$

Incorporating a hidden layer with 2 nodes using the linear activation function as well, we have that

$$y = w_0 + w_1\left(w_{0,1} + \sum_{i=1}^{2} w_{i,1}s_{i,1}\right) + w_2\left(w_{0,2} + \sum_{i=1}^{2} w_{i,2}s_{i,2}\right)$$

where w_0 denotes the intercept of the output neuron, $w_{0,1}$ *and* $w_{0,2}$ the intercept of the hidden layer nodes. w_1 and w_2 denotes the weights corresponding to the hidden layer neuron nodes and $s_{i,1}$ and $s_{i,2}$ represent the mean forecasts of OU and GBM models. These forecast are respectively modelled as follows:

$$s_{t,1} = s_{0,1}e^{-\theta t} + \mu(1 - e^{-\theta t})$$

$$s_{i,2} = s_{0,2}e^{\mu t}$$

To make forecasts using the neural network trained we use as inputs strictly pure diffusion models and pure jumps models respectively described by:

$$s_{t,1} = s_{0,1} + \sigma\varepsilon\sqrt{t}$$

$$s_{t,1} = s_{0,1} + jd\tilde{N}$$

Where $d\tilde{N}$ represents the Poisson process, ε is the Brownian motion process, σ and j are the volatility and jump sizes assumed to be equal.

3.7 System Development Methodology

According to Sommerville (2006), with Rapid Prototyping, also known as Rapid E-learning, learners or subject matter experts interact with prototypes and instructional designers in a continuous review and revision process. The rapid prototype creates an early iteration loop that provides valuable feedback on technical issues, creative treatment, and effectiveness of instruction.

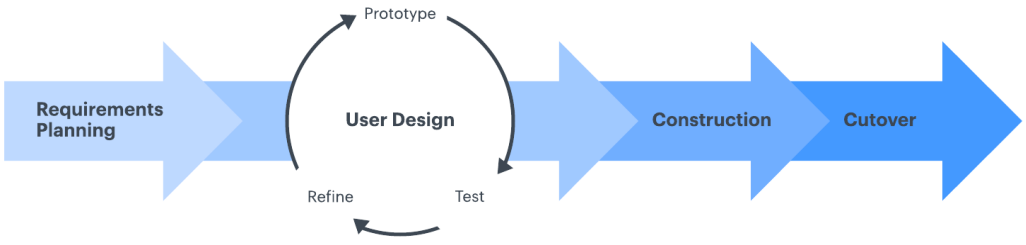

Figure 3.2: RAD Development Cycle

(Source: Sommerville, 2006)

How RAD Development Methodology was applied to the Research

(i) Requirements Planning- This involved getting the system requirements and doing a quick analysis. In this phase, all the tools and necessary materials were gathered, and the plan for the whole process was created.

(ii) Prototyping phase- In this phase, the designs and models for the prototype were created. The development followed thereafter.

(iii) Testing- This phase involved the validation of the models created. Unit, integration and system testing were done.

(iv) Cutover- This is the implementation phase where the finished product was launched. It involved data conversion, testing, and changeover to the new system, as well as user training. All final changes were made while the researcher and users sought for bugs in the system.

3.9 Programming Tools

The programming language that was used for the hybrid portfolio allocation prototype is python for the application development. The underlying forecasting algorithms and neural network algorithm were ran R but interfaced into python using the python wrapper rpy2 module. *Quadprog* and *neuralnetwork* R-based library was used within the prototype to perform the probability-based quadratic optimization given the estimated return and volatility value from the artificial neural network.

3.10 Ethical Considerations

In this study, in order to adhere to ethical codes of conduct, it was ensured that no sensitive personal information was collected from the users and analysis was done strictly on the data the user entered. Algorithms outputs were used as conducted by the algorithm and no modifications were done. The prototype had no access to user personal files and data. The results of the prototype were used to inform the portfolio selection process of investors by giving optimal portfolio allocation weights. This study received ethical clearance from Strathmore University Institutional Ethics Review Committee (SU-IERC) under approval number SU-IERC0599/19 and also received clearance from NACOSTI under license number NACOSTI/P/20/4388.

Chapter 4 : System Architecture and Design

4.1 Overview

After conducting a thorough review of literature on portfolio allocation and stock prediction techniques, and examination benefits and challenges of adopting a hybrid programming architecture for the predictive prototype. The following functional and non-functional requirements of the proposed hybrid portfolio allocation prototype were summarized.

4.2 Functional Requirements

The functional requirements were gathered through literature review and surveys of the stakeholders requirements. These functional requirements are descriptions of the inputs, behavior, services and output that the prototype will offer. The following are the functional requirements of the prototype:

a) The prototype should allow upload of user data in csv form with daily stock data of at most 5 stocks for a minimum duration of 1 year.
b) The prototype should allow the user to set a level target return of the portfolio, the expectation probability of the target return, the probability of zero-loss exceedance (VaR probability) and the duration of the investment.
c) The prototype should predict the prices of each stock using artificial neural networks given underlying OU and GBM mean models.
d) The prototype should present the predicted stock values used in portfolio selection to the user
e) The prototype should allow the user to download the predicted stock values in a csv file
f) The prototype should allow the user to create visualization of the predicted values using line plots
g) The prototype should use quadratic optimization to select the portfolio weights of each stock given inputs in (b) above
h) The prototype should present the portfolio weights to the user
i) The prototype should present the total computational time taken to complete computation
j) The prototype should allow the user to go back and load a different set of data in csv form

k) The prototype should allow the user to save the results

l) The prototype should allow the user to retrieve the results of a previous session

4.3 Non-functional Requirements

4.3.1 Usability

The Graphical User interface of the prototype should be easy to use and to navigate.

4.3.2 System Performance and Reliability

The hybrid prototype adopts a multilanguage programming paradigm as can be used to leverage of the latency of Python and powerful R based libraries.

4.3.3 System Scalability and Modification

The neural networks algorithm can be scaled to have more hidden layers and input nodes. The optimization algorithms should be scalable to allow an additional number of assets beyond the currently supported 5 assets.

4.3.4 Portability

The prototype should be portable to various hardware platforms based on Windows Operating System.

4.4 System Architecture

The system architecture is described in Figure 4.1. It describes the multilanguage programming approach that will be used where the algorithms will be developed in a hybrid computing environment with the base algorithms of prediction using artificial neural networks and quadratic optimization algorithm are ran in R and the development of the prototype is done within Python to leverage on Python speed and application development utilities.

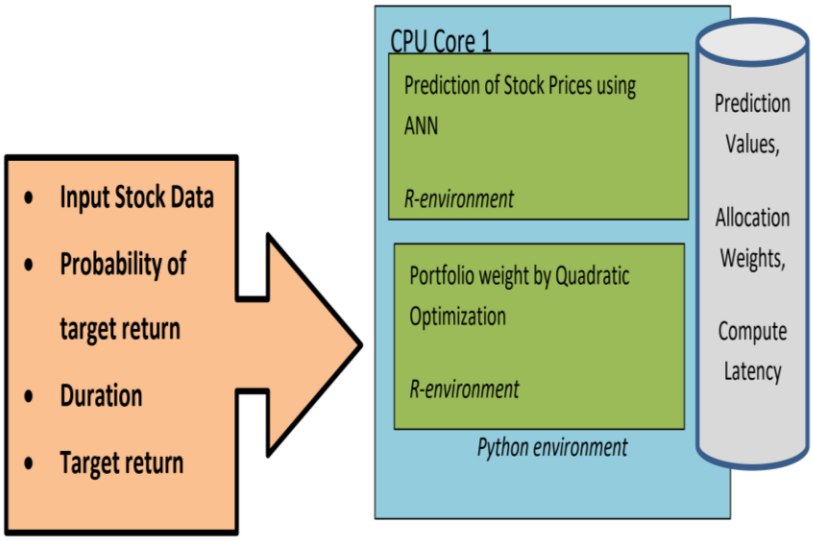

Figure 4.1 : System Architecture Diagram

This prototype design is suited to ensure that the user inputs *csv* form data and these input stock variables are used to perform the prediction and subsequently the portfolio allocation. First the user gathers the data of the stocks they would like to invest in, then inputs the probability expectations of the target return, the duration and the probability of zero-loss exceedance. The prototype shall then make use of the R based neural networks algorithms and the quadratic optimization algorithms. The prediction, weights and computation time data once a session has been performed shall be stored within a database within the prototype and could be exported to a *csv* file. This system architecture allows efficient development of the application of a multilanguage paradigm that would ensure optimal optimization algorithm performance as well as application performance.

4.5 System Design

The design modelling of the portfolio allocation prototype was done using UML (Unified Modelling Language). The tools for modelling used are use case diagrams, sequence diagrams, data flow diagrams, activity diagrams and entity relationship diagrams.

4.4.1 Use Case Diagram

Use case diagrams show the interactions between the system and its external actor. It is used to model requirements and contexts of a system. Use case diagrams are composed of use cases,

actors, associations and a system boundary. Each use case describes a requirement that exists in the system such as input csv data, input target return, perform allocation. These requirements are shown to relate to other requirements or actors using associations. An association describes a relationship between two diagram elements that represents communication or interaction between them. The interaction between Actors and the proposed prototype have been described in Figure 4.2.

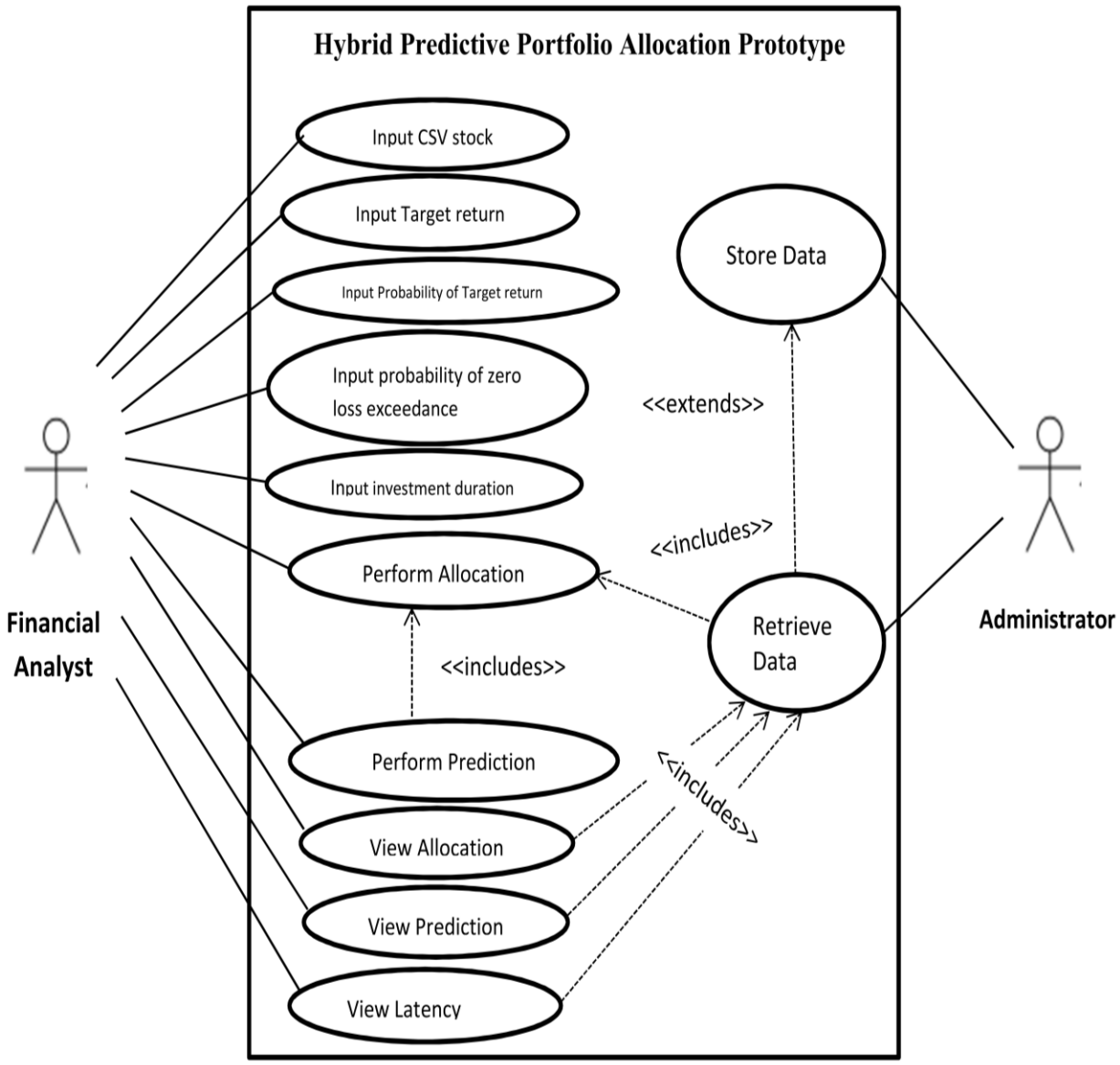

Figure 4.2: Use case diagram of portfolio allocation prototype

4.4.2 Entity Relationship Diagram

Figure 4.3 describes the entity relationship diagram showing the relationship between the entities within the lightweight database. The system's entities of portfolio requirements, stock and results have a one to many and many-to-many relationship as illustrated. Results from the prototype can arise out of many different portfolio requirements specifications and also many stock data entries.

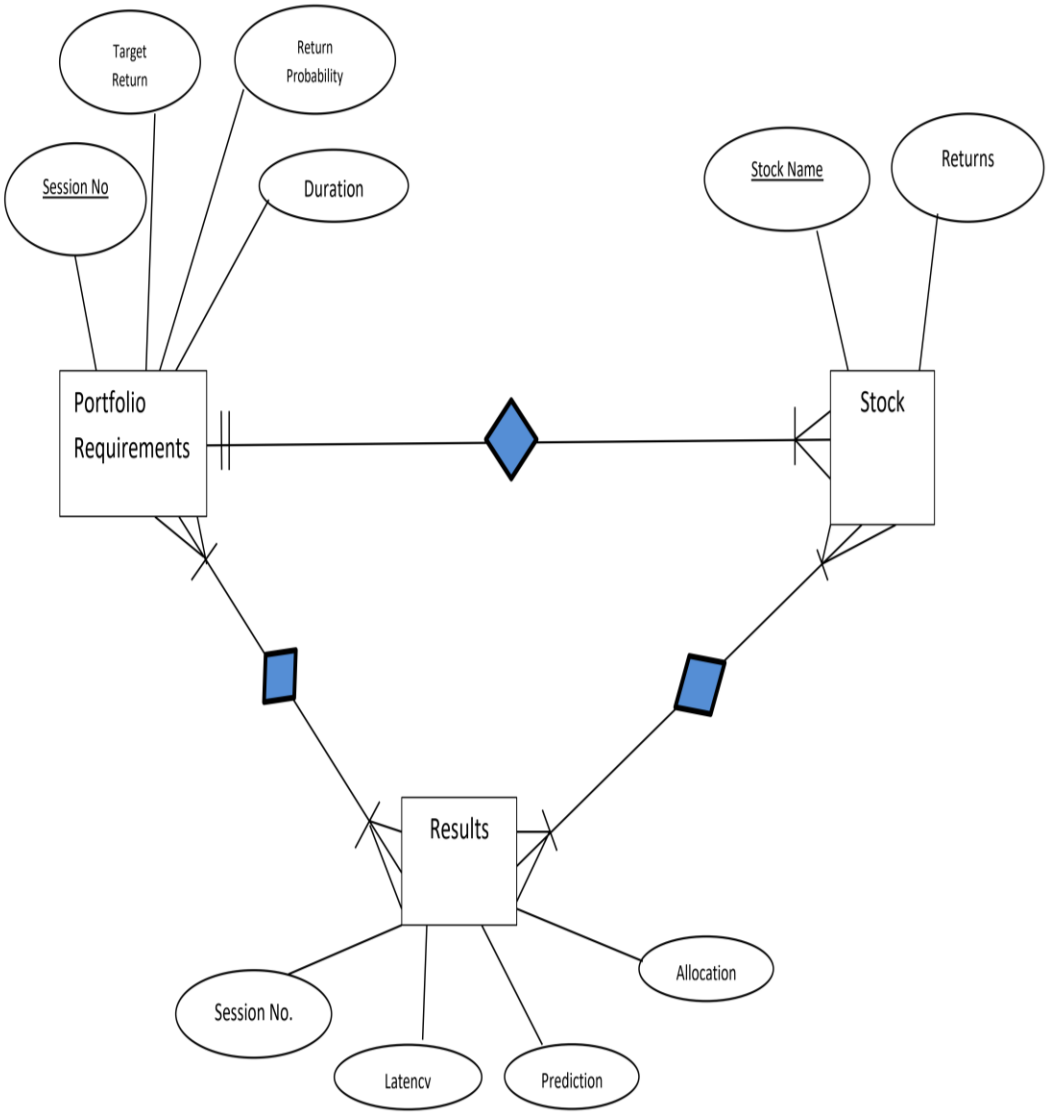

Figure 4.3: Entity Relationship Diagram

4.4.3 Data Flow Diagram

The data flow diagram in Figure 4.4 represents the flow of a data from processes in the system prototype. The financial analyst would be able to input the data and the variables. These inputs would then be used in the performing of predictions using ANN and subsequent performing of allocations using the quadratic programming algorithm.

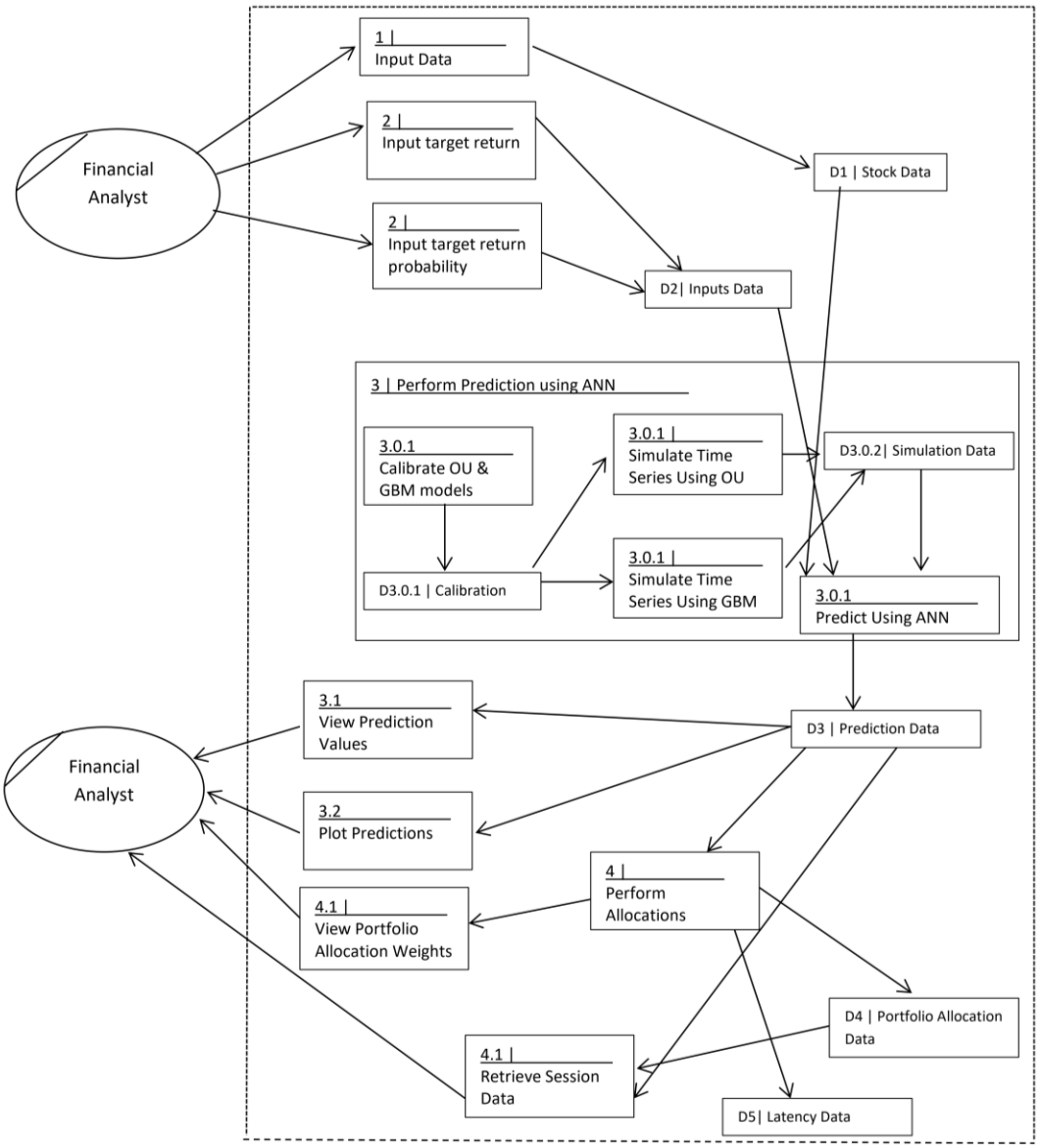

Figure 4.4: Data Flow Diagram

4.4.4 Sequence Diagram

Figure 4.5 presents the sequence diagram which describes the interaction between the prototype system objects in a sequential manner depicting the flow and duration of execution. The financial analyst inserts data which is in CSV form. The data is the store in the SQLite database. To perform the allocations and the prediction the data is then spilt into two the training and test datasets. The OU and GBM prediction estimated are used as inputs into the ANN model and prediction of stock prices are obtained. Returns are then computed and fed into the probability quadratic optimization algorithm to perform the allocations. The results are then saved. The analyst can retrieve these results from the database and further view plots of the ANN and time series plots of the data and predictions.

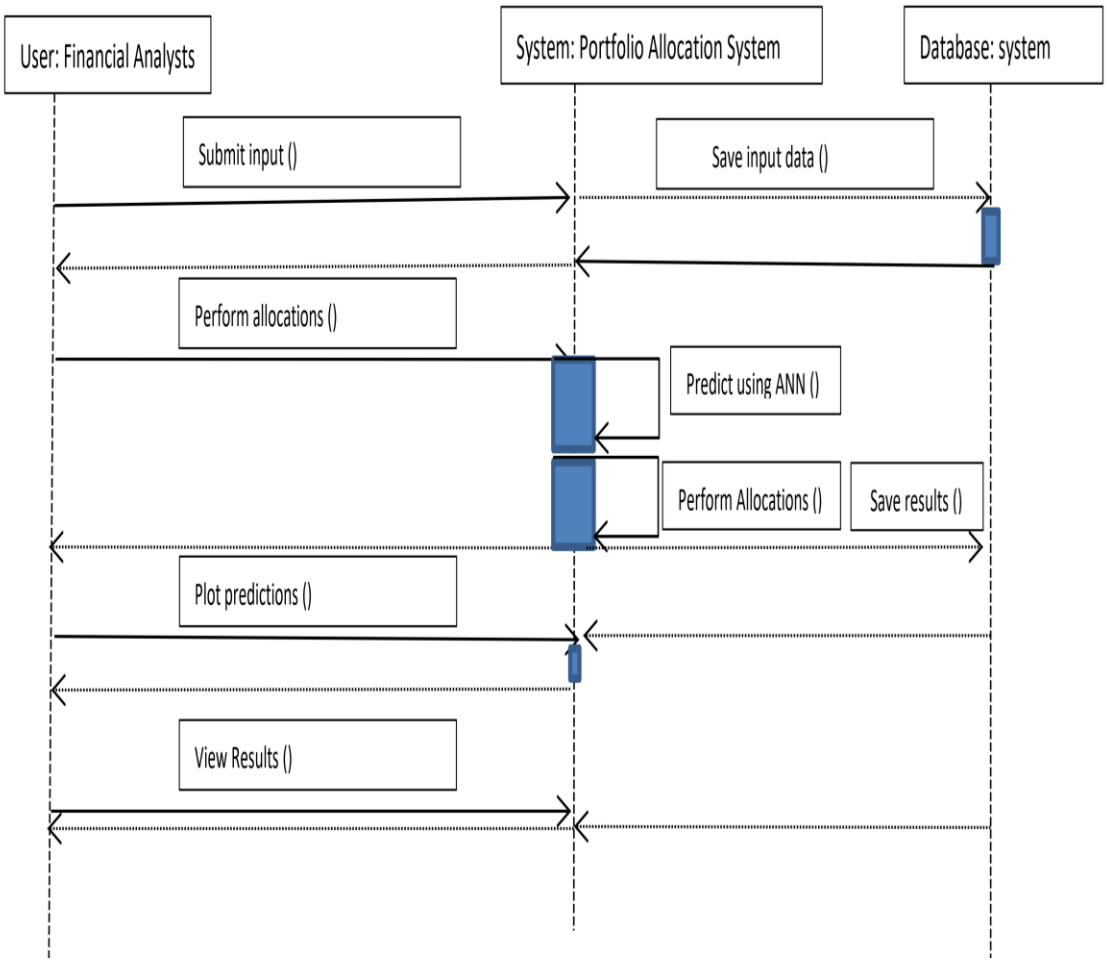

Figure 4.5: Sequence Diagram

4.4.5 User Interface Design

Figure 4.6 describes the user interface for the hybrid predictive prototype. The user is required to input values of target probability, level of zero-loss exceedance probability and target portfolio return. He can then click on the function he requires the hybrid predictive prototype to perform.

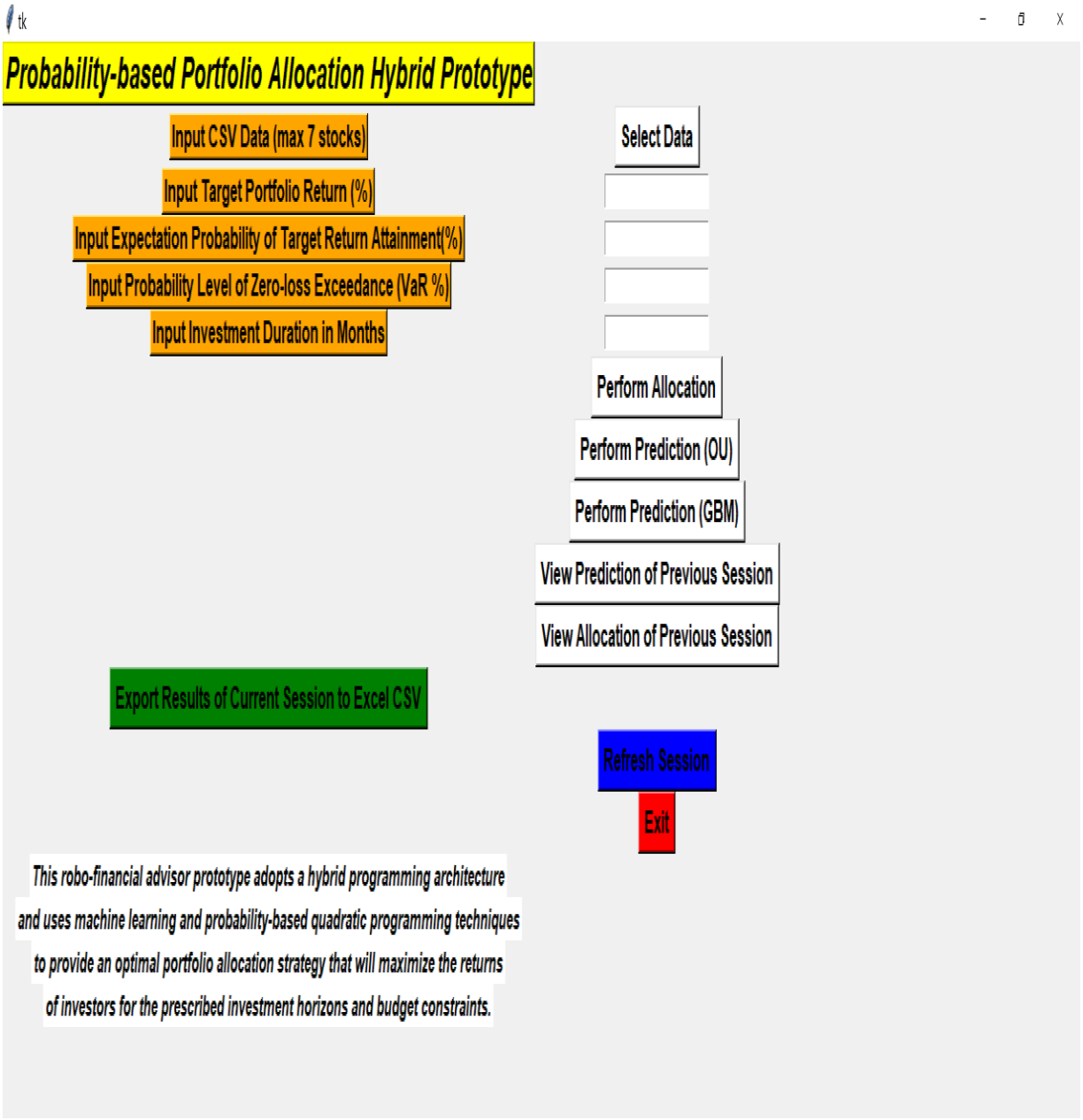

Figure 4.6: User Interface Design

Chapter 5 : Implementation and Testing

5.1 Overview

This chapter illustrates the implementation, testing and results from the predictive portfolio allocation prototype. The prototype was 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 artificial neural network were price estimated by OU and GBM models, and output of actual prices was used to train the artificial neural network. The data entered used was of 5 stocks for a period of 5 years to train the artificial neural network.

5.2 System Requirements

5.2.1 Hardware Requirements

The hardware requirements for running the prototype to obtained optimal portfolio allocations have been summarized in Table 5.1.

Table 5.1: Hardware Requirements

Hardware	**Minimum Requirements**
Processor	i5 3rd Generation (used i7 8th Generation)
Number of Processor Cores	8
Cycle Speed	2.5 GHz
Hard Disk Space	250 GB
RAM	4 GB
Graphic Card	Intel, AMD, Nvidia
Display Resolution	1080x1920

5.2.2 Software Requirements

The software requirement to run the hybrid predictive portfolio allocation prototype have been summarized in Table 5.2.

Table 5.2: Software Requirements

Software	Minimum Requirements
Operating System	Windows 7 (Windows 10 Pro Used)
Anaconda Python	Version 3.7
R	Version 3.6
Windows Excel	2010 Suite
Database	SQLite

5.3 Implementation of Prototype

The Hybrid Predictive Prototype was developed using a hybrid programming approach in Python and R, where the artificial neural network algorithms and quadratic optimization algorithms were written in R and interfaced into Python using the library rpy2. This integration enabled us to leverage on the benefits of each programming language, the R computational libraries and Python application development and speed.

5.3.1 Data Inputs

The data inputs that were required for the prototype to perform predictions and allocations were the dataset in CSV form of 5 assets for a period of at least 1 year. A dataset that was obtained from Yahoo Finance of a period of 5 years was used. The prototype also required the following data inputs namely, the level target return, the expected probability of target return, the probability of zero loss exceedance and the duration.

5.3.2 Neural Network Algorithm

The prototype implemented the artificial neural network algorithm in R and interfaced it into Python using the rpy2 interface package. The artificial neural network used 4 hidden layers. The output layer consisted of the training dataset of the stock prices of the 5 assets. The input layer consisted of estimates of the stock prices by OU and GBM models. The outputs of this algorithm were the predicted stock prices for the desired duration of the user and the plots of the fitting of the artificial neural network estimates.

5.3.3 Probability Quadratic Optimization Algorithm

The prototype implemented this algorithm in R and interfaced it into Python using the rpy2 interface package. This algorithm provided the various allocation weights for the assets given the input parameters of the level target return, the expected probability of target return, the probability of zero loss exceedance and the duration. This algorithm also provided the portfolio risk (standard deviation) that will be attained by the portfolio for the given duration with the obtained allocation weights.

5.3.4 Software Integration

The prototype architecture integrated R-libraries and functionality within Python environment, and algorithms were estimated in this hybrid environment. The prototype database used to save predictions of stock prices and portfolio allocations was SQLite. The prototype also integrated with Microsoft Excel by allowing use of data in CSV form to be uploaded and used in the prediction and allocation of portfolio weights. Also, saved data of predictions and weights could be extracted and save separately as a CSV file. Once the hardware and software requirements of the computing device have been met, the prototype software was installed in .exe form and files extracted. The prototype worked in Windows OS.

5.3.5 Interface Design

The graphical interface was designed to be easy to use and quite direct where the user is meant to input data in a csv file and then input the level target return, the expected probability of target return, the probability of zero loss exceedance and the duration. Once this data has been input into the prototype, then the user can select which function he wants to perform. He can select to perform predictions where the estimation values of the OU and GBM will be produced and the artificial neural network will be estimated. The user can also select to perform allocations in this function the prototype will use the probability quadratic algorithm to give the weights of the five assets and the portfolio standard deviation (portfolio risk). This perform allocation function includes estimation of the neural network.

5.3.5.1 Perform Predictions

The results of estimation of the OU and GBM inputs into the neural network algorithm by the prototype as summarized in the the Figure 5.1. These were then used as inputs in the artificial

neural nework and subsequent results we used to obtain the portfolio allocations by the probability based quadratic optimization algorithm.

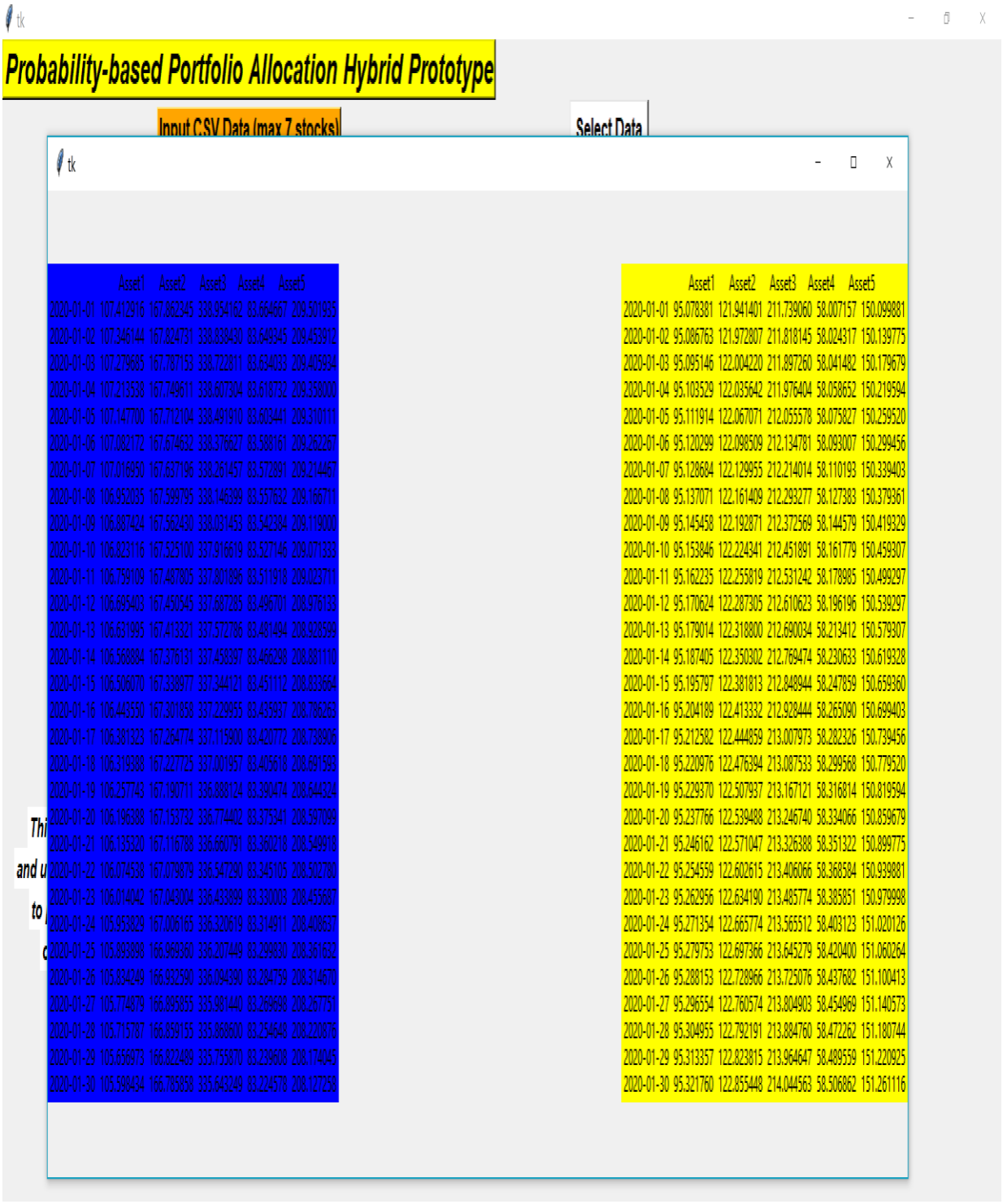

Figure 5.1: Prediction Time series results from Prototype

5.3.5.2 Perform Allocations

The results from the probability quadratic optimization were obtained by the prototype and have been shown in Figure 5.2. The Figure 5.2 shows the portfolio weights and the portfolio standard deviation (portfolio risk) for a portfolio of 5 assets.

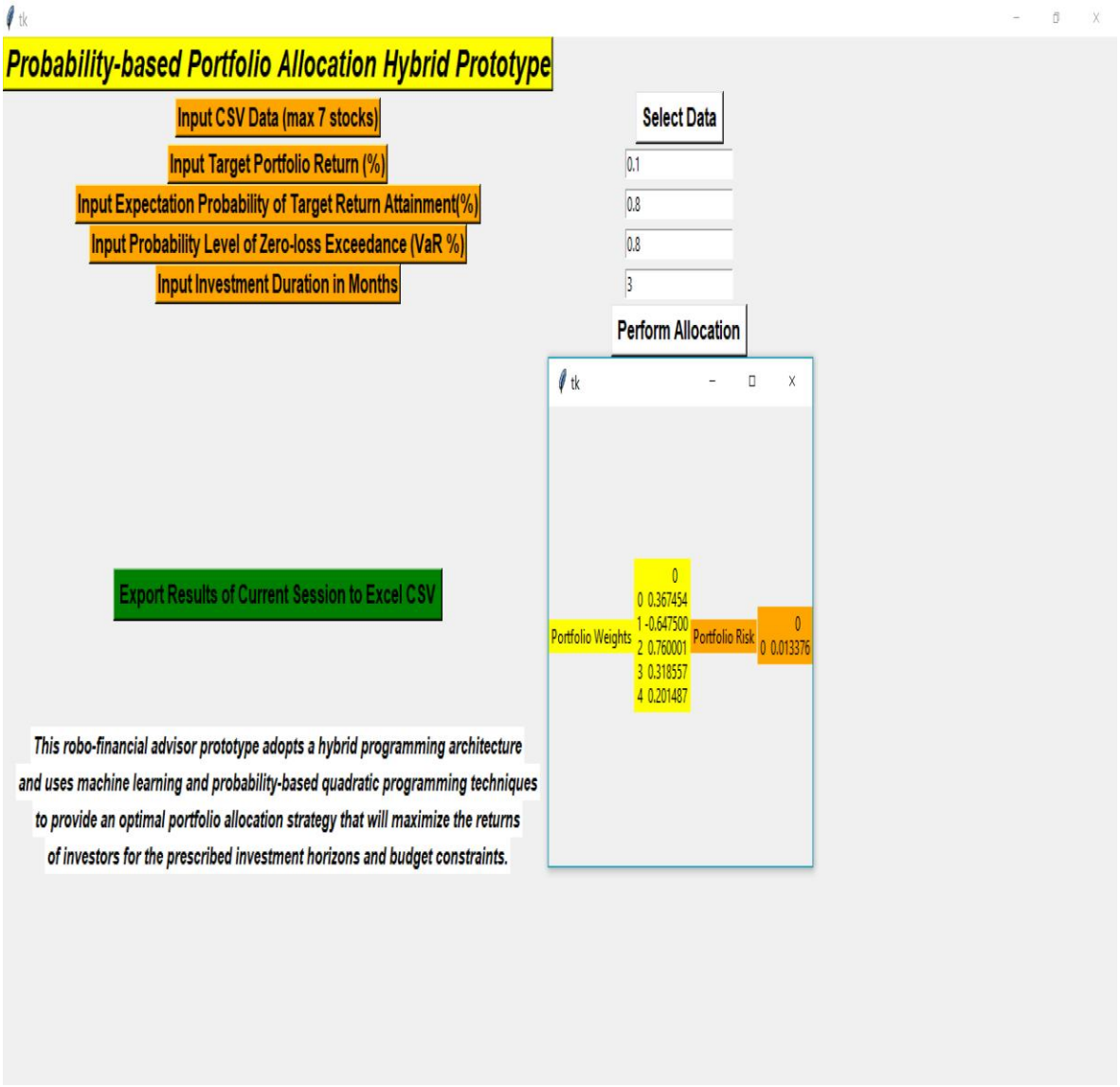

Figure 5.2: Portfolio Allocation Weights from Prototype

5.4 System Testing

The system testing was done by using different combination of assets of maximum 5, and summarizing the portfolio allocation weights and the computational times. This was redone to check the robustness of the neural network prediction algorithms and the portfolio algorithms. The prototype functionality was also tested by checking whether it meets all the functional and non-functional requirements. The results on the functionality and non-functional requirements tests have also been summarized in Table 5.3. The results from these algorithm robustness tests have been summarized in Table 6.1.

Table 5.3: Functional and Non-function Requirements Testing

No.	Action	Results	Outcome
1.	Upload of user data in csv form with daily stock data of at most 5 stocks	To be updated finalizing of tests	Pass
2.	Input a level target return of the portfolio	Users were able to input the target return level in decimal form	Pass
3.	Input expectation probability of the target return	Users were able to input the probability of target return in decimal form	Pass
4.	Input the probability of zero-loss exceedance	Users were able to input the probability of zero loss exceedance in decimal form	Pass
5.	Input duration of the investment	Users were able to input the duration of the investment in months or fraction of months	Pass
6.	Summary Prediction values by ANN	Prototype provided summary of the prediction values by the artificial neural networks for a 3 month duration	Pass
7.	Download the predicted stock values in a csv file	Predicted stock values could be extracted to a csv file	Pass
8.	Create visualization of the predicted values using line plots	Time series plots of the predicted values were create	Pass
9.	Obtain portfolio weights by quadratic optimization	Prototype was able to perform portfolio allocation using the quadratic optimization algorithm	Pass
10.	Summarize portfolio weights to the user	The portfolio allocation weights were summarized, allowing strictly positive weights	Pass
11.	Obtain total computational time	The prototype provided a summary of the total computation time in running the algorithms to obtain the results	Pass
12.	Navigate prototype to load a different set of data in csv form	We were able to create a different dataset of 5 assets and obtained the artificial neural network predictions and the portfolio allocation	Pass
13.	Save the results of sessions	Results were stored in SQLite database after every portfolio allocation computation	Pass
14.	Retrieve the results of a previous session	Users were able to retrieve results of immediate previous session	Pass
15.	Prototype is portable to various hardware platforms using different Windows OS	Prototype could be installed and used in a similar way on another Windows OS device	Pass
16.	Graphical User Interface is easy to use and navigate	User interface was direct input-submit-results form without any complexities in the design	Pass

Chapter 6 : Discussion

6.1 Overview

The aim of this research was to develop a robo-financial advisor prototype which adopts a hybrid programming architecture and uses machine learning techniques to provide an optimal portfolio allocation strategy that maximizes the returns of investors for the prescribed investment horizons and budget constraints. This prototype utilized a weighting of prediction algorithms by using artificial neural networks to predict the expected price of assets in the portfolio. The underlying inputs in the prediction algorithm were multi-asset OU and GBM of models. It then applied probability-based quadratic optimization algorithm to provide the optimal portfolio allocation strategy that maximizes on investors returns in the given investment horizon. We present the results of these algorithms in this section, achievement of objectives and challenges.

6.2 Model Implementation Outputs

The following are presentations of results in Table 6.1, Figure 6.1 and Figure 6.2. The results of implementation of the prediction algorithm and the portfolio allocation algorithm in the prototype using a dataset with 5 assets and having varying inputs of portfolio target return , portfolio zero loss exceedance (value at risk level), portfolio target return level, and the different durations. These results have been summarized in the Table 6.1.

Table 6.1: Results of Stock Allocation (Asset 1-RWR, Asset 2 – IWM, Asset 3- EFA, Asset 4 – FM, Asset 5 –GSPC)

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)	[Asset 1] [Asset 2] [Asset 3] [Asset 4] [Asset 5]	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)	[Asset 1] [Asset 2] [Asset 3] [Asset 4] [Asset 5]	[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
3*E(R)		NA, NA	NA, NA	NA, NA	NA, NA
Medium Term (2 months), 1*E(R)	[Asset 1] [Asset 2] [Asset 3] [Asset 4] [Asset 5]	[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
3*E(R)	3	NA, NA	NA, NA	NA, NA	NA, NA
Long Term (6 months), 1*E(R)	[Asset 1] [Asset 2] [Asset 3] [Asset 4] [Asset 5]	[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
3*E(R)	3	NA, NA	NA, NA	NA, NA	NA, NA

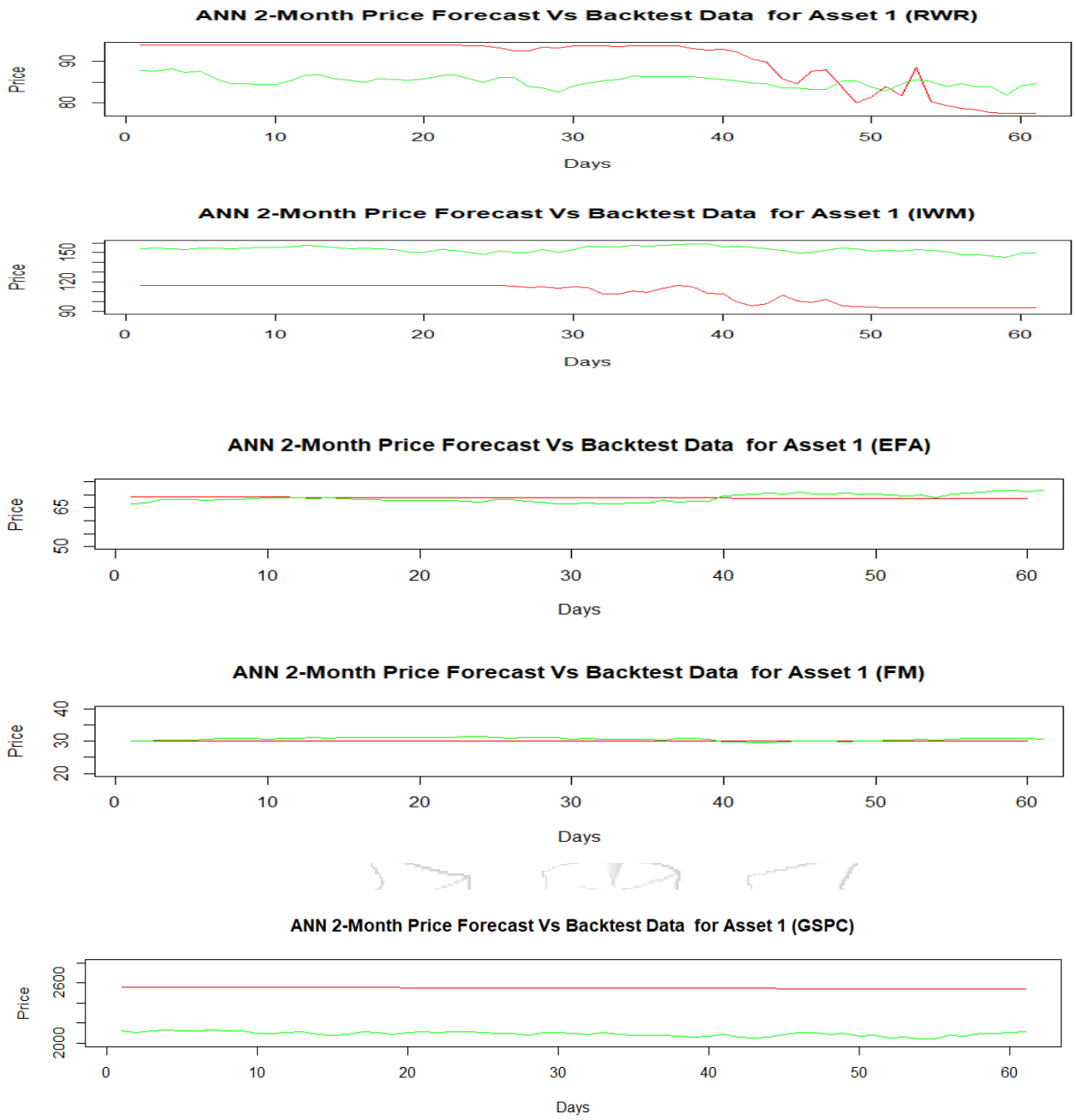

Figure 6.1: Time series Plot of Predicted Stock Prices

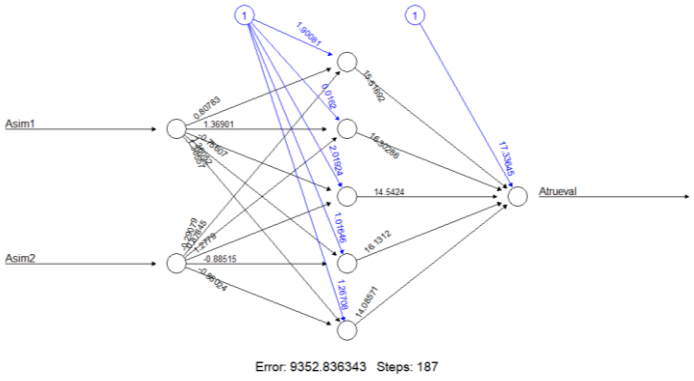

Figure 6.2(a)

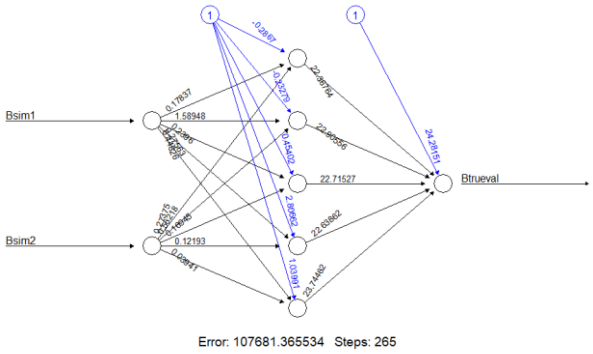

Figure 6.2(b)

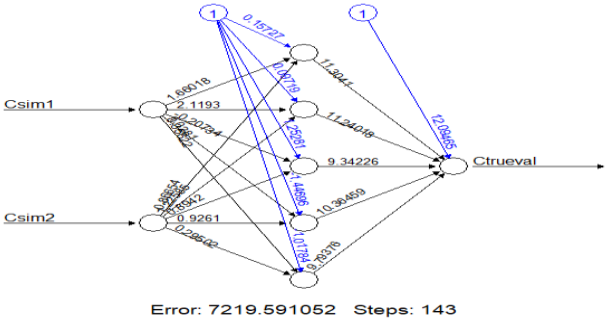

Figure 6.2(c)

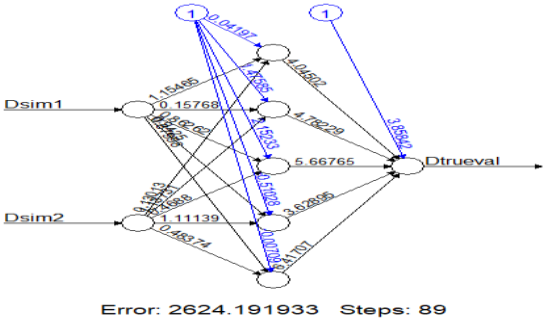

Figure 6.2(d)

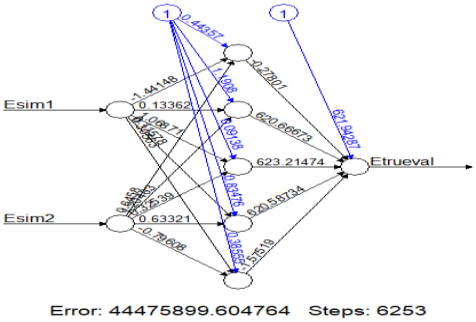

Figure 6.2(e)

Figure 6.2: Artificial Neural Network from training models (OU, GBM) and training data (Stocks RWR, IWM, EFA, FM and GSPC respectively-labelled (a) to (e))

6.3 Algorithms Results

From the prototype algorithms tests we find that the quadratic programming algorithm using the neural network forecasts could not converge at high expected return levels to yield portfolio weights. The algorithm however converged at very low expected return levels to yield the portfolio weights. The returns produced by the neural network forecasting algorithm we very low close to zero consistently. The OU and GBM forecasting models produced slightly higher returns. Thus the quadratic programming algorithm was found to perform well when forecast returns had a high standard deviation. When the quadratic programming algorithm was tested with historical data and OU forecasts and it also performed generally well, but however could not converge when implied target return was generally high (factors of 2,3 above implied return). These results of the allocation weights obtained by the quadratic programming algorithm using ANN predictions are shown in Table 6.1.

Generally, the neural network algorithm performed averagely in forecasting as shown in Figure 6.1 depicting the 2-month neural network forecasts against the historical test data. Predicting accurately the level of 2 of the assets (EFA and FM) and also prediction correctly the trend of 2 of the assets (RWR and IWM). The neural network was found to give a convergent price forecast when more hidden layer nodes (more than 5) where applied and returns with more layers were yielding zero returns consistently. These neural network algorithm was found to be much more suitable for short-term (30 day forecasts) rather than longer term (1 year forecast) in the latter it performed poorly, the prices had very low close to zero standard deviation and consequently very low returns. OU and GBM models forecasts had higher standard deviations than the artificial neural network forecasts.

Asset 4(FM) in all algorithm test cases had received the highest allocation weight, except when the expected return level was doubled. The lowest variance portfolio was a medium term portfolio with variance of 0.003378745 (0.334%). Its expected target return was however -0.045% at 95% target return confidence. The highest variance portfolio was a short term portfolio with variance of 0.01542699 (1.542699%). It expected target return was 0.0056 (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.

6.4 Results Summary

The first objective of this study was to analyze factors relating to portfolio selection. This was address by conducting a comprehensive literature review to establish the factors that inform portfolio allocation and found that the portfolio returns, duration and the portfolio standard deviations were the main factors that would be considered in determining the portfolio allocation. This summary was presented in Chapter 2. The techniques that were applied in existing portfolio allocation platforms were investigated and it was found out that none adopted neural network prediction techniques together with quadratic optimization. The Markowitz mean-variance approach was the most commonly used in existing robo-advisors for portfolio allocation. We suggested in Chapter 3 an algorithm that could combine the prediction of stock prices for an investment duration using neural networks to achieve accurate predictions whereby the inputs to the neural network model were stock price estimates from OU and GBM. This neural network prediction results were then fed into a probability based quadratic optimization algorithm that took into account the duration of investment, the probability of achieving a certain return, the probability of achieving a loss more than zero from the portfolio and the individual stocks standard deviation in order to obtain portfolio allocation weights that will yield a portfolio with low risk (standard deviation). The results of these algorithms were presented in Chapter 6.1. The developed prototype performed very well in estimating the portfolio allocations and performing predictions. The algorithm could however not converge at certain input levels such as when the expected return level was very high or individual stock returns were consistently zero. The results on the functionality and design of the prototype have been presented in Chapter 5.

Chapter 7 : Conclusions and Recommendations

7.1 Summary of Conclusions

The first objective of this study was to analyze factors relating to portfolio selection. This was address by conducting a comprehensive literature review to establish the factors that inform portfolio allocation and found that the portfolio returns, duration and the portfolio standard deviations were the main factors that would be considered in determining the portfolio allocation. The techniques that were applied in existing portfolio allocation platforms were investigated and it was found out that the Markowitz mean-variance approach was the most commonly used in existing robo-advisors for portfolio allocation. An portfolio allocation algorithm that could combine the prediction of stock prices for an investment duration using neural networks to achieve accurate predictions whereby the inputs to the neural network model were stock price estimates from OU and GBM was suggested in Chapter 3. In its development the resulting neural network prototype would make use of these artificial neural network and quadratic programming algorithms to perform portfolio selection. It took 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 prototypes algorithms allowed asset allocation in a portfolio with a maximum of 5 assets. The neural network algorithm performed averagely well forecasting being able to predict the correct level of 2 of 5 assets and to predict correctly the trends of the remaining 3 assets, as shown in the time series plots in Figure 6.1. We however noted that the neural network prediction produced prediction with very low standard deviations as compared to the input OU and GBM models. This is expected as it purposed to rely these models as the training models, therefore recurrently optimized the weights on the nodes. This shows that the artificial neural network approach to forecasting could be a reliable technique to forecasting albeit at short duration of about 30 days since at longer horizons and greater number of hidden nodes the algorithm seems to be static even with heavy shocks to the underlying inputs.

The quadratic optimization algorithm was found to be sensitive to the level of the target return selected. If the target returns were very high or if individual stock returns were consistently zero, then the algorithm would not converge to yield optimal weights. Setting of the target return within twice the actual historical portfolio return, but not higher, was found to lead to

convergence and give optimal results. This algorithm supported investment in shorter time horizons since portfolio risk was lowest. The best well-diversified portfolio allocation was achieved in these shorter time horizons. Longer horizons allocations were biased towards asset with lower standard deviations. Lowest risk portfolios were the ones with a lower certainty probability of target return and highest risk portfolios were the ones with a higher certainty probability of target return. Generally, the quadratic algorithm supported high allocations of above 50% to Asset 1(RWR) and Asset 4 (FM) both within the short term. Thus from this selected portfolio the robo-advisor prototype system suggests that it would be more profitable to invest in RWR and FM in the short term horizon although one would have exposure to high volatility of the portfolio.

7.2 Recommendations and Future Research

The results on the functionality and design of the prototype were also analyzed and presented. They showed 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 prototype was able to use, within Python, advanced R-based quadratic optimization, calibration, and neural network libraries for portfolio analysis that were 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 to improve the performance of the prototype. A historical approach to portfolio optimization could also be considered where the historical returns are strictly adopted with the quadratic problem.

References

Abraham, F., Schmukler, S. L., & Tessada, J. (2019). Robo-advisors: Investing through Machines. *World Bank Group: Research and Policy Briefs*.

Balasubramanian, J., Mintz, A., Kaplan, A., Vilkov, G., Gleyzer, A., Kaplan, A., et al. (2010). Adaptive Parallel Computing for Large Scale Distributed and Parallel Applications. *ACM*.

Beketov, M., Lehman, K., & Wittke, M. (2018). Robo Advisors: Quantitative Methods Inside the Robots. *Journal of Asset Management*, 363-370.

Bekkerman, R., Bilenko, M., & Langford, J. (2011). *Scaling Up Machine Learning - Parallel and Distributed Approach.* Cambridge University Press.

Bjernes, L., & Vukovic, A. (2017). Automated Advice: A Portfolio Management Perspective on Robo-Advisors. *Masters Thesis: Norwegian University of Science and Technology*.

Broadie, M. (1992). Computing Efficient Frontiers Using Estimated Parameters. *Annals of Operations Research*.

Business Insider. (2017). *Robo-Advisors: online Financial Advisors that Fit in Your Pocket.* Business Insider.

Chopra, V. (1993). Improving Optimization. *The Journal of Investing*.

Chopra, V., & Ziemba, W. (1993). The Effect of Errors in Means, Variances and Covarinace on Optimal Portfolio Choice. *Journal of Portfolio Management*.

Deloitte. (2016a). *The Expansion of Robo-advisors in Wealth Management.* London: Deloitte White Paper.

Deloitte. (2016b). *Robo Advisory in Wealth Management.* London: Deloitte White Paper.

Djehiche, Y. (2018). A Neural Networks Approach to Portfolio Choice. *KTH Royal Institute Working Paper*.

Freitas, F. D., Souza, A. F., & Gomes , F. J. (2009). Portfolio Selection with Predicted Returns Using Neural Networks. *Federal University of Espirito Santo Working Paper*.

Grothey, G. (2009). Parallel Computing: Financial Applications. In R. Trobec, *Parallel Portfolio Optimization.* Springer.

Hatemi, J., & El-Khatib, Y. (2015). Portfolio Selection: An Alternative Approach. *Economics Letters*, 424-427.

Lam, J. W., & Swensen, A. (2016). Robo-Advisors: A Portfolio Management Perspective . *Yale University Thesis*.

Markowitz, H. (1952). *Portfolio Selection: Efficient Diversification of Investment.* New York: Wiley.

Michaud, R. (1989). The Markowitz Optimization Enigma: Is 'Optimized' Optimal? *Financial Analysts Journal*.

Niio. (2018). *Robo-Advisory in the Securities Market: A Guide to Opportunity Assessment and Implementation.* NIIIO SQUARE GROUP.

Prigent, J. (2007). *Portfolio Optimization and Performance Analysis.* Chapman & Hall.

Renaux, M. C., Rudnicki, M., Beniere, A., & Lorain, F. (2019). *How will Robo-advisors Reshape Asset Management?* Initio Square Group.

Singh, R. H., Barford, L., & Harris Jr, F. (2016). Accelerating the Critical Line Algorithm for Portfolio Optimization using GPUs. *Information Technology: New Generations*.

Sommerville, I. (2006). *Software Engineering (7th Edition).* Addison Wesley.

Statista. (2018). *Robo Advisors.* Statista Online Portal : Worldwide Market Study.

Swensen, D. F. (2009). *Pioneering Portfolio Management.* Free Press.

Verelst, L. (2019). *Robo-advisor Model Renewal: Quantitative Models, Dynamic Risk Assessment and Customer Centricity.* Initio Square Group.

Appendix

A. SU-IERC Ethical Clearance

Strathmore UNIVERSITY

27th February 2020

Mr Muganda, Brian
bw.muganda@gmail.com

Dear Mr Muganda,

RE: A Decentralized Predictive Portfolio Selection Prototype

This is to inform you that SU-IERC has reviewed and **approved** your above research proposal. Your application approval number is **SU-IERC0599/19.** The approval period is **27th February, 2020 to 26th February, 2021.**

This approval is subject to compliance with the following requirements:

i. Only approved documents including (informed consents, study instruments, MTA) will be used
ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-IERC.
iii. Death and life threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-IERC within 72 hours of notification
iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-IERC within 72 hours
v. Clearance for export of biological specimens must be obtained from relevant institutions.
vi. Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.
vii. Submission of an executive summary report within 90 days upon completion of the study to SU-IERC.

Prior to commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology and Innovation (NACOSTI) https://oris.nacosti.go.ke and also obtain other clearances needed.

Yours sincerely,

f/: Dr Virginia Gichuru,
Secretary; SU-IERC

STRATHMORE UNIVERSITY INSTITUTIONAL ETHICS REVIEW COMMITTEE (SU-IERC)
27 FEB 2020
TEL: +254 (0)703 034 000
P. O. Box 59857 - 00200
NAIROBI - KENYA

Cc: Prof Fred Were,
Chairperson; SU-IERC

B. NACOSTI Research Permit

REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: **311848**

Date of Issue: **25/March/2020**

RESEARCH LICENSE

This is to Certify that Mr.. Brian Wesley Muganda of Strathmore University, has been licensed to conduct research in Nairobi on the topic: A Parallel GPU Accelerated Predictive Prototype for Portfolio Selection using Probability-based Quadratic Programming and Neural Networks for the period ending : 25/March/2021.

License No: **NACOSTI/P/20/4388**

311848

Applicant Identification Number

Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.