Artificial Neural Networks in Stock Return Prediction: Testing Model Specification in a Global Context

By

Naa Ayorkor Buxton-Tetteh
BXTNAA002

Submitted to the University of Cape Town
In partial fulfilment of the requirements for the degree

Master of Commerce specialising in Finance in the field of Investment Management

Department of Finance and Tax
Faculty of Commerce
University of Cape Town

Supervisor: Professor Paul van Rensburg
August 2019

Declaration

I, N.A. Buxton-Tetteh, hereby declare that this thesis is entirely my own work. Each significant contribution to, and quotation in, this thesis from the work, or works of other people has been attributed, and has been cited and referenced. This thesis is not being submitted for degree purposes at any other university.

Signed: Naa Ayorkor Buxton-Tetteh
20 August 2019

Acknowledgement

I would like to acknowledge and thank my supervisor, Professor Paul van Rensburg, for his guidance and assistance through the completion of this thesis. His valuable comments and suggestions are greatly appreciated.

I would also like to thank God, my family and my friends for their love and endless support throughout this journey. This would not have been possible without you all.

Abstract

This research investigates whether artificial neural networks which make use of firm-specific fundamental and technical factors can accurately predict the returns of a sample of several large-cap stocks from various markets across the globe. This study also explores which hidden layer configuration leads to the best network predictive performance. Furthermore, this research identifies which firm-specific factors predominantly influence the predictions made by the artificial neural networks.

Five artificial neural networks are designed, trained and tested on a sample of 161 stocks from the Russell 1000 and the S&P International 700 stock indices. The investigation period extends over a 166-month period from January 2001 to October 2014 with a 70:30 split for training and testing subsamples respectively. Eighteen firm-specific factors, based on prior research about the presence of style effects or anomalies on the cross-section of global equity returns, are used as the input variables of the artificial neural networks to forecast one-month forward returns of all the stocks in the sample.

The five artificial neural networks investigated in this research differed in hidden layer size. Specifically, the number of hidden neurons examined were three, nine, 13, 18 and 30. All five networks train significantly well, with each network's training error indicating a good model fit. Each network also achieves the desirable information coefficient of 0.1 between its predicted returns and the actual returns in the training sample. It is interestingly discovered that network performance generally improves as the number of hidden neurons in the hidden layer increases until a specific point, after which network performance weakens.

In the context of avoiding overfitting, the best-trained network in this research is that with 13 neurons in its hidden layer. This is the primary network used for the out-of-sample testing analysis. This network achieves an average prediction error magnitude of approximately 7% and an information coefficient of 0.05 during out-of-sample testing. These results underperform their respective benchmarks moderately. However, further analyses of the network's performance suggest an overall poor out-of-sample predictive ability. This is illustrated by a significant bias and a considerably weak relationship between the network's predicted returns and the actual returns in the testing sample.

Global sensitivity analysis reveals that growth style effects, particularly, the capital expenditure ratio, return on equity, sales growth, 12-month percentage change in non-current assets and six-month percentage change in asset turnover were the most persistent factors across all the ANN models. Other significant factors include the 12-month percentage change in monthly volume traded, three-month cumulative prior return and one-month prior return. An unconventional result of this analysis is the relative insignificance of the size and value style effects.

Table of Contents

List of Tables

List of Figures

List of Terms and Acronyms

ANN	Artificial Neural Network
APT	Arbitrage Pricing Theory
CAPM	Capital Asset Pricing Model
EMH	Efficient Market Hypothesis
MAD	Mean Absolute Deviation
MLP	Multilayer Perceptron
S&P	Standard & Poor's
SSE	Sum of Square Errors

1 Introduction

1.1 Introduction

The purpose of this research is to investigate the use of artificial neural networks for predicting stock returns in the global equity market. This chapter introduces the thesis and sheds light on the research problem, objectives and contributions to existing literature. The outline of this chapter is as follows: Section 1.2 presents some background and context to the study; Section 1.3 describes the research problem and the associated research questions; Section 1.4 discusses the objectives of the research; Section 1.5 highlights the significance of the research and its contribution to existing literature; Section 1.6 concludes the chapter and proceeds to describe the structure of the study and the outline of the following chapters.

1.2 Background

The Efficient Market Hypothesis (EMH) is a theory which states that investors should not be able to outperform the market as security prices already incorporate and reflect all relevant historical, firm-specific and insider information (Fama, 1970). Passive investing is based on the belief of efficient markets and intends to match market performance through replication strategies. Conversely, active investing stems from the belief that there are existing mispriced securities as a result of market inefficiency.

There is an ongoing global debate surrounding active versus passive investment management. Promoters of active investing assert that the market presents mispricing opportunities that can be exploited to generate excess returns. On the other hand, advocates of passive investing argue that these are simply market anomalies whose benefits are short-lived as a result of active management and performance fees. In light of this, there has undeniably been a substantial amount of evidence mounting up for the case against active management (Fama and French, 2010).

Notwithstanding this evidence, fundamental and technical analysis, which are two of the major techniques used by active investment managers when selecting securities that have potential to outperform the general market, are still utilised. The motivation for using fundamental and technical analysis stems from the presence of firm-specific style effects or anomalies and price trends observed in the cross-section of equity returns. Fundamental analysis involves thorough research into firm-specific factors that inform the stock's intrinsic value. These firm-specific factors are obtained from a firm's financial statements and financial ratios. Technical analysis relies heavily on historical stock price data and trading activity as indicators of future stock price performance.

Rapidly evolving advancements in statistical computing are presenting new and improved techniques that will change the traditional way in which the financial sector operates. In particular, artificial neural networks (ANNs), which form a subset of machine learning algorithms, are becoming increasingly popular in the field of finance. For practitioners and researchers, ANNs are proving to be effective statistical modelling tools in areas such as pattern classifications, pattern recognition and forecasting.

Artificial neural networks are modelled on the biological neural networks found in the human brain. ANNs are able to learn and self-train from data at their disposal and as a result, can capture and postulate both linear and non-linear functional relationships between variables. A valuable characteristic of ANNs is their lack of reliance on assumptions around distributional properties and independence which most traditional statistical methods require.

Given the magnitude and the prevalent accessibility of historical information, ANNs can serve as practical models to boost efficiency and accuracy in the prediction of stock returns. This research aims to investigate if ANNs can be applied effectively to predict the returns of a broader sample of stocks in the global equity market.

1.3 Research Problem

Research into the use of ANNs in finance has gradually emerged in the last two decades - according to Wong and Selvi (1998), the earliest publication dates back to 1990. Majority of the published research to date has seen the application of artificial neural networks in areas such as bankruptcy predictions of firms and banks, and predictions of stock price performance. However, there is limited research investigating the use of ANNs in global stock return prediction.

This research therefore aims to answer the following core research question:

"Can artificial neural networks that employ firm-specific inputs accurately predict future stock returns?"

Furthermore, there is no formal procedure which guides the design of an artificial neural network's architecture, especially its hidden layer. Although there are existing heuristics in place, the specification of this network parameter relies on a certain degree of experimentation. Consequently, this research will attempt to answer the following research sub-question:

"What hidden layer size leads to an optimal network forecasting performance?"

The core research question stated above will be answered in reference to the global equity market. The ANN model will be applied to a sample of commonly held stocks listed on the Russell 1000 Index and the S&P International 700 Index. Prior literature on fundamental and technical factors that influence global stock prices and returns will be consulted in order to ascertain which variables to include as inputs in the ANN. Further analyses of ANN performance, by means of global sensitivity analyses, will attempt to answer the following research sub-question:

"Which firm-specific inputs provide the most significant information for the return predictions made by the artificial neural networks in this study?"

1.4 Research Contribution

This aim of this study is to contribute to research around the use of artificial neural networks in directly predicting stock returns for individual stocks. In particular, this study will focus on model specification to investigate which types of artificial neural network architecture, if any, achieve strong predictive performance. This research will also make use of firm-specific technical and fundamental explanatory variables that have been suggested to influence stock returns in the past to examine whether these suggested relationships persist with ANN models. Furthermore, prior research around ANNs has largely focused on specific markets such as the U.S., Taiwan and Japan. This research therefore aims to explore the topic on a new and broader sample.

1.5 Outline of the Research

This research paper consists of 7 chapters. This chapter has introduced the research problem and has provided the background to the study. Chapter 2 presents the theoretical background of this research, which includes theory of artificial neural networks. This chapter will also present literature on the Efficient Market Hypothesis. Chapter 3 presents existing literature on the general usefulness and limitations of artificial neural networks as well as prior research on the use of ANNs in the field of finance. Chapters 4 and 5 discuss the data and the methodology that will be used to build the ANN models for analysis in this research. Chapter 6 presents the research findings and analysis thereof. Chapter 7 concludes the research paper, highlights the limitations of the study and proceeds to make recommendations for future research.

2 Theoretical Background

2.1 Introduction

Fadlalla and Lin (2001) describe artificial neural networks as "computerized intelligence systems that simulate the inductive power and behaviour of the human brain". ANNs are able to emulate the neural processes which occur in the biological human brain. Analogous to the human brain, ANNs are information processing units that are able to learn, generalize and propose relationships from information at their disposal (Svozil, Kvasnicka and Pospichal, 1997). These key features promote the use of ANNs for modelling purposes especially as non-linear alternatives to mainstream linear modelling methods.

The efficient market hypothesis postulates that in an efficient market, a stock's price should reflect all the available information pertaining to the stock (Fama, 1970). This implies that an investor transacting based on the information accessible in the market should not be able to generate excess risk-adjusted returns (Jensen, 1978). The efficient market hypothesis presents itself in three different forms namely, weak form, semi-strong form and strong form. Each form of the hypothesis proposes a different set of information which determines market efficiency.

This chapter presents the theoretical background which underpins this research. This includes an introduction to the concept of artificial neural networks and their use as non-linear modelling systems in Section 2.2. The concept of market efficiency and the various forms of the Efficient Market Hypothesis are also reviewed in Section 2.3. The final section, Section 2.4, provides a summary and concludes the chapter.

2.2 Artificial Neural Networks

2.2.1 The Biological Neuron

The biological human brain is made up of approximately 10 billion nerve cells called neurons. There are an estimated 60 trillion synapses or connections between these neurons of the brain (Haykin, 1994). These neurons operate in interconnected clusters which form what is formally known as a neural network.

Figure 2.1 below depicts the structure of two connected biological neurons found in the human brain.

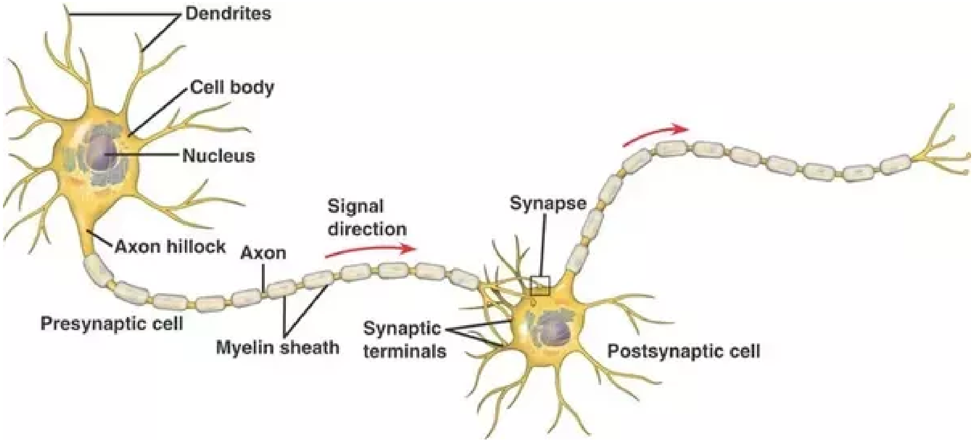


Figure 2.1: Two Connected Biological Neurons, (Moya, 2011)

Each biological neuron is made up of a cell body, which is surrounded by hair-like figures called dendrites, and contains the cell nucleus. Each neuron also comprises a long tail-like structure called the axon and terminal buttons at the end of the axon. The terminal buttons of one neuron connect to the dendrites of the next neuron through a connection called the synapse (Haykin, 1994). This serves as the most common channel through which information is transferred from one neuron to another.

Once a stimulus is received from the external environment, receptors convert it into an electrical impulse. The receptors subsequently channel the impulse into the brain and into the neurons via their dendrites. The information contained in the stimulus is subsequently evaluated through the neurons and synapses. An output signal is then released from the system to the effectors and is converted into responses i.e. either decisions or actions (Nygren, 2004). In the biological neural network, learning takes place through the fluctuation of synapse responsiveness which affects the impression that one neuron has on another neuron (Siganos and Stergiou, 1996).

2.2.2 The Artificial Neuron

The underlying framework of an artificial neuron is derived from the biological neuron. The first artificial neuron model was developed in 1943 by McCulloch and Pitts from their knowledge of neurology. Their model was notably simplistic, consisting of binary inputs, i.e. values of either 0 or 1 , identical weights, a fixed threshold activation function and a single binary output (McCulloch and Pitts, 1943). Several simplifying assumptions and constraints were imposed in the construction of their model (McCulloch and Pitts, 1943). Post the McCulloch and Pitts model, there have been several advancements in the artificial neuron model. Figure 2.2 below depicts the typical artificial neuron (Nygren, 2004; Ashwood, 2014).

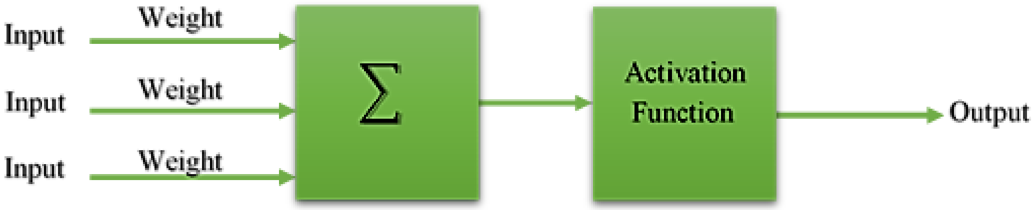


Figure 2.2: An Artificial Neuron

The typical artificial neuron has the following components:

- Input variables;
- Scalar weights which are applied to each input variable;
- A summation function;
- An activation function;
- An output variable;
- A learning algorithm.

The input variables are equivalent to stimuli which are directed into the artificial neuron via their respective weights. The scalar weights, which are either positive or negative, imply the strength of their corresponding input variables. The summation function calculates the weighted sum of the input factors and channels this into the non-linear activation function which is responsible for defining how the various inputs combine to produce the output variable.

The typical artificial neuron can now take on a range of inputs factors and weights. It can also make use of various activation functions and can produce output variables which are not necessarily binary. By virtue of this, the artificial neuron model has been freed from the limiting assumptions and constraints underlying the McCulloch and Pitts model (Jain, Mao and Mohiuddin, 1996; Ashwood, 2014).

Advancements of the model have also led to the introduction of a learning mechanism. This enables the artificial neuron to modify the weights of each input until the desired output is achieved. Figure 2.3 shows the artificial neuron with the learning mechanism (Ashwood, 2014).

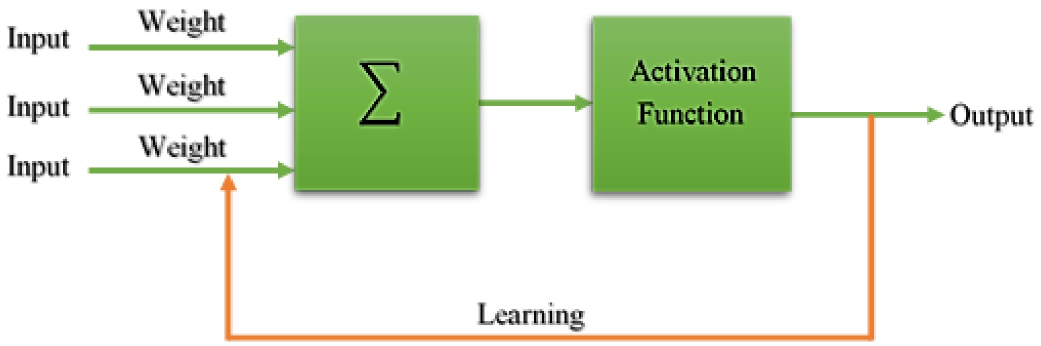


Figure 2.3: The Artificial Neuron with Learning Mechanism

An artificial neuron can also be represented mathematically as follows (Shachmurove and Witkowska, 2000):

$$y = f(z)$$

$$z = \sum_{i=1}^{n} x_i w_i$$

Where:

- x_i are the input variables;
- w_i are the scalar weights;
- n is the total number of input variables;
- z is the weighted sum of the input variables;
- $f()$ is the activation function and;
- y is the output variable.

It is helpful to consider the artificial neuron on similar grounds as its counterpart, linear regression. The input factors for an artificial neuron can be regarded as the independent or explanatory variables in a linear regression model while the output variable corresponds with the dependent variable. The scalar weights for the input factors are akin to the coefficients of the linear regression model.

2.2.3 Activation Functions

Prior to the application of the activation function in the artificial neuron, the weighted sum of the input factors is computed. The activation function is then implemented on the resulting computation to generate an output from the artificial neuron. The activation function produces output variables which often fall in the ranges of [0, 1] or [-1, 1] (Shachmurove and Witkowska, 2000; Nygren, 2004). Although a variety of activation functions exists, the three most frequently used are the following (Jain, Mao and Mohiuddin, 1996; Nygren, 2004; Siganos and Stergiou, 1996; Ashwood, 2014):

1. **Threshold function**

   The threshold function is typically applied when a binary output is required. The function takes on the following form:

$$f(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$

2. **Piecewise linear function**

   The piecewise linear function is given by the following equation:

$$f(x) = \begin{cases} 1 & x \geq \frac{1}{2} \\ x & -\frac{1}{2} < x < \frac{1}{2} \\ 0 & x \leq -\frac{1}{2} \end{cases}$$

3. **Sigmoid function**

   The sigmoid function takes on two forms: the hyperbolic tangent sigmoid function and the logistic sigmoid function. The former is used to generate output values between -1 and 1 while the latter is used to generate output values between 0 and 1. Their respective equations are given below:

- Hyperbolic tangent sigmoid function:

$$f(x) = \tanh(x)$$

- Logistic sigmoid function:

$$f(x) = \frac{1}{1 + e^{-x}}$$

The distinguishing feature which differentiates the artificial neuron from the typical linear regression model is non-linearity which is often captured in the activation function of the artificial neuron. While the activation function for the artificial neuron is typically non-linear, the activation function for a linear regression model is the identity function which takes on the form $f(x) = x$ (Hodnett, 2010).

2.2.4 Artificial Neural Network Architecture

An ANN consists of numerous interconnected artificial neurons. The architecture of the network is generally structured into layers of artificial neurons which can ultimately take on three arrangements:

1. **Single-layer feed-forward network**

   This is the most basic form of a neural network. The input neurons, which form part of the input layer, are received by the network and directly processed into the neurons in the output layer. The input layer is not computational and is therefore not considered as a formal layer. Due to the feed-forward nature of the network, there are no feedback loops present. The figure below depicts an example of a single-layer feed-forward neural network with three input neurons and two output nodes.

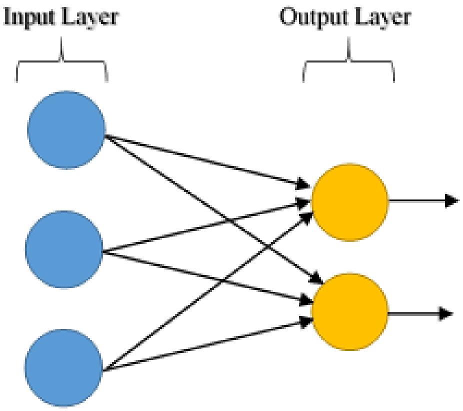


Figure 2.4: A Single-layer Feed-forward Neural Network

2. **Multilayer feed-forward network**

   A multilayer feed-forward network, also known as a multilayer perceptron (MLP) (Jain, Mao and Mohiuddin, 1996), is formed when one or more additional computational layers are added to the single-layer feed-forward network. These layers are called hidden layers and fall between the input layer and the output layer of the network. The hidden neurons in the hidden layers serve as additional information processing units which precede the final computation of the output variables. The presence of hidden layers also improves the network's capacity to develop higher-order statistics from the input neurons (Haykin, 1994). Figure 2.5 below shows an example of an MLP 3-2-1 network. This is a multilayer feed-forward neural network with three input neurons, two hidden neurons in one hidden layer and one output neuron.

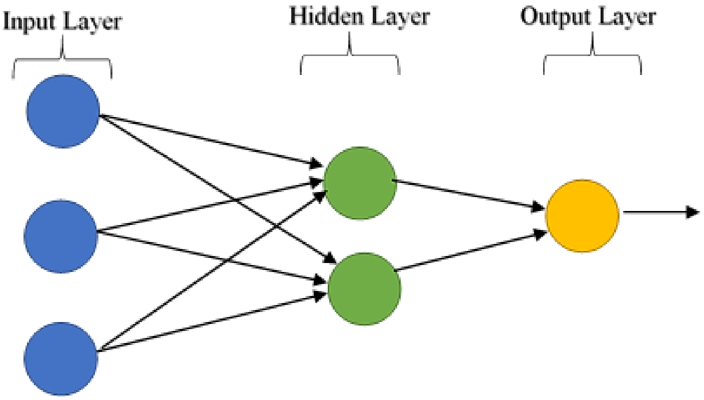


Figure 2.5: A Multilayer Feed-forward Neural Network

3. **Recurrent Networks**

Unlike feed-forward network structures where stimuli can only travel forward, recurrent networks have at least one feedback loop which provides the network with feedback from every round of computation (Jain, Mao and Mohiuddin, 1996). Figure 2.6 provides an illustration.

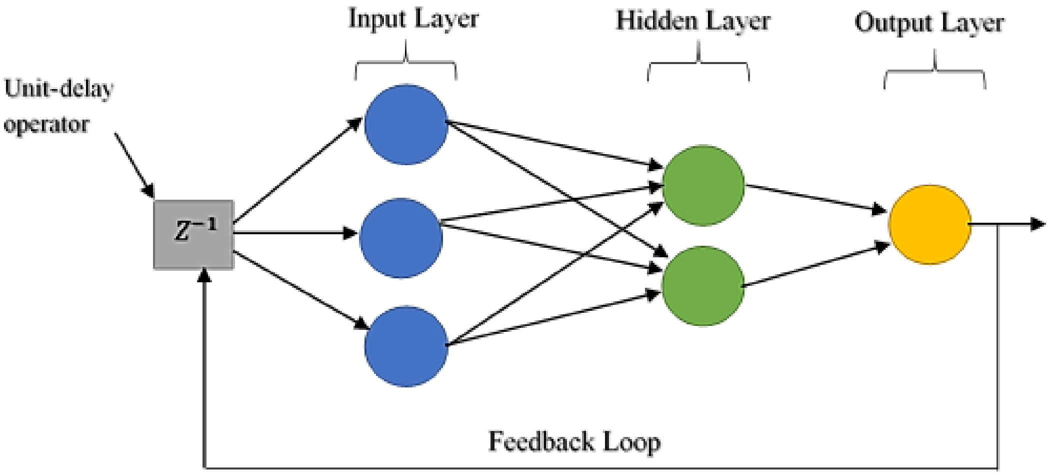


Figure 2.6: A Recurrent Neural Network

The linear regression model can be considered as a single-layer feed-forward network with one output neuron and the identity activation function in its output layer.

2.2.5 Artificial Neural Network Training

In order for an ANN to develop the required competency to solve problems, it needs to be trained. The training or learning process is centred on determining the most accurate weights for the input factors that will result in the computation of the correct output. The general process involves feeding the network with training data, allowing the network to compute output values, comparing these computed outputs to the target output from the training data and updating the input weights using the information contained in the discrepancy or error between the computed and target outputs. This process is iteratively performed until the ANN produces significantly accurate outputs.

There are three main methods through which ANNs learn: supervised learning, unsupervised learning and hybrid learning. Supervised learning entails feeding the network with input data and desired output data. The ANN is then responsible for defining the appropriate input weights that will enable it to produce output values that are closely accurate to the desired output values. Unsupervised learning, on the contrary, does not require the provision of desired output data. The ANN has the freedom to explore the input data, postulate relationships and patterns in the data and ultimately propose its own output variables in response to the input data. Hybrid learning, as the name suggests, is a blend of supervised and unsupervised learning. In this case, the input weights are determined by both supervised and unsupervised learning methods (Jain, Mao and Mohiuddin, 1996).

The backpropagation algorithm, which will be the training method used in this research, was originally developed by (Werbos, 1974). This algorithm falls under the supervised learning category of neural network training. It functions most suitably with a multilayer feed-forward network structure with at least one hidden layer.

The backpropagation algorithm uses the delta rule which is a gradient descent training method (Buscema, 1998). This rule requires the activation function in each computational neuron to be bounded and differentiable (Ashwood, 2014). The algorithm operates by evaluating the error or discrepancy between the computed output from the network and the target output from the data. The error is then circulated back into the network and is used to update the input weights. This process repeats iteratively until the error between the network's computed output and the desired output is minimised (Buscema, 1998).

The process of backpropagation can be demonstrated mathematically. The error, E_j, between the computed output Y_j and the target output O_j can be illustrated as follows (Buscema, 1998):

$$E_j = \frac{1}{2}\sum_j (O_j - Y_j)^2$$

As previously discussed, the network's computed output Y_j, is a function of the input factors and their respective weights. The gradient descent method updates the input weights as follows (Buscema, 1998):

$$w_{ij+1} = w_{ij} - \eta\frac{\delta E}{\delta w_{ij}}$$

Where,

- w_{ij+1} is the i^{th} weight's value in the $(j+1)^{th}$ iteration;
- w_{ij} is the i^{th} weight's value in the j^{th} iteration;
- η is the learning rate;
- $\frac{\delta E}{\delta w_{ij}}$ is the partial derivative of the error with respect to weight w_{ij} which can be broken up further using the chain rule of differentiation into $\frac{\delta E}{\delta Y_j} \times \frac{\delta Y_j}{\delta w_{ij}}$.

2.3 Efficient Market Hypothesis

The foundational definition of an efficient market is a market in which security prices fully incorporate and reflect all pertaining information which is available (Fama, 1970). According to Jensen (1978), "a market is efficient with respect to information set θ_t if it is impossible to make economic profits by trading on the basis of information set θ_t". In other words, an investor should not be able to generate any excess risk-adjusted returns, after costs, from trading based on the information available in the market (Jensen, 1978).

In general, the efficient market hypothesis (EMH) adopts three different forms: *weak form*, *semi-strong form* and *strong form* (Fama, 1970).

The *weak form* of market efficiency asserts that all security prices in the market fully reflect any information contained in past price history (Fama, 1970). According to this form of the EMH, technical analysis has no justification as it should not produce any excess returns if markets are weak form efficient. In his paper, Fama (1970) asserts that if markets are efficient in the weak form then the daily changes in prices should follow a random walk as future prices should be independent of prior prices. Any evidence of serial correlation between price changes and therefore returns, would disprove this form of market efficiency (Fama, 1970).

The *semi-strong form* of the EMH suggests that prices in the market fully reflect all information that is publicly available, including any historical price information (Fama, 1970). This also includes firm-specific information contained in financial statements, news reports and any other material and publicly available information. Semi-strong market inefficiency is tested through observing the speed at which security prices adjust to reflect public announcements (Fama, 1991). Any evidence that asserts that certain firm-specific factors have forecasting ability on future returns disproves semi-strong market efficiency (Fama, 1991).

The *strong form* of market efficiency asserts that security prices in the market fully reflect all available information (Fama, 1970). This includes all private information in addition to historical price and public information. This form of market efficiency renders insider trading unjustified (Fama, 1970). However, there is empirical evidence suggests that this form of market efficiency rarely holds as there is private information that is not fully incorporated in prices (Fama, 1991).

Although this outline of the EMH was initially proposed by Fama in 1970, Fama later suggested alternative definitions of these three forms of market efficiency in his 1991 paper. The *weak form* category was altered to include a more extensive concept of *tests for return predictability* (Fama, 1991). For the other two categories, Fama (1991) proposed alterations to their titles and not to their underlying concepts. The *semi-strong form* category was changed to have a more informative title of *event studies* and the *strong form* category was also altered to have a more informative title of *tests for private information* (Fama, 1991).

As Grossman and Stiglitz (1980) suggest, the prerequisite for the EMH to hold is that all relevant costs (trading and information) are always zero. Fama (1991) acknowledges that this is hardly the case in practice. By virtue of this, a more reasonable variation of the EMH would postulate that prices incorporate information up until the marginal benefits from having access to the information is equivalent to the marginal cost of attaining the information (Jensen, 1978; Fama, 1991).

For this research, if an artificial neural network model can successfully predict the returns of the largest stocks across the global equity market using publicly available firm-specific indicators, this would present some evidence against the EMH, particularly the semi-strong form. The inverse is also true – if an artificial neural network model cannot effectively predict stock returns, there would be evidence supporting the EMH.

2.4 Conclusion

This chapter has reviewed the theoretical foundations of this research. In particular, the concept of ANNs has been explored in-depth. The EMH has also been reviewed extensively and has been considered in the context of this research.

ANNs are inspired by the neural networks that are found in the human brain. A typical ANN is made up of several interconnected artificial neurons which act as information processing units within the network. The artificial neuron consists of input variables, scalar weights for each input variable and output variables. An activation function is also present in an artificial neuron and is responsible for establishing how the different input variables combine to produce the output variable. The activation function of the artificial neuron is often non-linear. This characteristic distinguishes the model from the standard linear regression which is considered to have the identity function as its linear activation function.

ANNs can adopt one of three main topologies, namely, a single-layer feed-forward, a multilayer feed-forward or a recurrent network design. These different architectures often influence the learning and training algorithm for the ANN. The backpropagation algorithm is a popular supervised learning method used specifically for training a multilayer feed-forward network. This algorithm applies the gradient descent rule during training to minimise the degree of error between the network's computed output and the desired output originally presented to the network. The process ultimately determines the ideal weights which best define the relationship between the input variables and the output variables.

The EMH, which takes on three different forms, asserts that in an efficient market, security prices should wholly reflect all available information (Fama, 1970). However, if there is any indication that an artificial neural network which uses publicly available firm-specific information can accurately predict stock returns, there would be reasonable evidence against the theory.

3 Literature Review

3.1 Introduction

The capital asset pricing model (CAPM) and the arbitrage pricing theory (APT) are classical asset pricing models which have been used for decades in an attempt to explain the cross-section of stock returns, predict stock prices and assist in portfolio selection. Several researchers have also developed alternative models for similar purposes as a result of the inability of these traditional models to capture asset pricing anomalies observed in the market. Underpinning all these models is multiple regression. The concept of multiple regression relies on the assumption of linearity to propose accurate relationships between a number of explanatory variables and a dependent variable.

ANN models have evolved as compelling alternatives to standard multiple regression models. Although the theories behind ANNs and multiple regression notably differ, several studies have found evidence which suggests that ANNs perform as well as, and sometimes better than, multiple regression models (Ashwood, 2014). ANNs models have been found to overcome some of the limitations that multiple regression analyses present and in doing so, provide attractive modelling advantages. Furthermore, these alternative modelling techniques have been used in various fields of research and notably in the field of finance.

This chapter reviews prior literature around the efficiency of ANN models and the use of such models in finance. Section 3.2 explores the usefulness and limitations of ANNs while Section 3.3 reviews prior research findings from the use of ANN in the field of finance. Section 3.4 concludes the chapter.

3.2 The Usefulness and Limitations of Artificial Neural Networks

A traditional statistical analysis method such as multiple regression depends on several assumptions about the variables used in the modelling process. Furthermore, the accuracy of this method relies on these assumptions being fulfilled (Osborne and Waters, 2002). The table below discusses the main assumptions of multiple regression and draws attention to some of the limitations encountered if these assumptions are not met (Osborne and Waters, 2002; Williams, Grajales and Kurkiewicz, 2013).

Table 3.1: An overview of the assumptions of multiple regression

Assumption	Explanation	Limitation
Linearity	Multiple regression assumes a linear relationship between the independent, explanatory variables and the dependent variable(s) of the model.	If non-linearity exists in the relationship between the variables, multiple regression will usually under-estimate the true relationship leading to inaccurate results.
Normality	In multiple regression analysis, it is assumed that the model errors are normally distributed.	The true relationship can potentially be misrepresented as a result of fat-tailed or highly-skewed errors. These non-normality characteristics render results untrustworthy.
Homoscedasticity	Multiple regression also assumes that the model errors exhibit variance that is constant across all levels of the independent, explanatory variables.	If the variance of the errors is inconsistent across the levels of the explanatory variables, heteroscedasticity is exhibited. A considerably large presence of this can lead to inefficient and unreliable results.

Alongside these assumptions, other factors such as multicollinearity (the existence of significant correlations between independent variables) and outliers are areas of concern for the modelling process in multiple regression analysis (Williams, Grajales and Kurkiewicz, 2013). Although artificial neural network models differ significantly from multiple regression models, they are often proposed as alternative modelling techniques (Uysal and El Roubi, 1999). This is particularly because ANN models overcome the limitations and concerns presented by multiple regression models and therefore offer more desirable modelling features (Uysal and El Roubi, 1999).

Zhang, Patuwo and Hu (1998) describe artificial neural networks as being self-adaptive and data-driven models. These features enable them to derive underlying relationships between variables present in a dataset even when these relationships are unknown or difficult to identify. Due to their learning ability, ANNs are able to generalize the information acquired from training sample data and then accurately infer on unseen data (Zhang, Patuwo and Hu, 1998). The authors note that although ANNs present these desirable characteristics, their development is nontrivial and requires careful configuration of all the model parameters (Zhang, Patuwo and Hu, 1998).

Another advantageous feature of ANNs is their non-linear framework. This feature facilitates their ability to capture non-linearity inherent in the relationships between the input factors and the output variables (Svozil, Kvasnicka and Pospichal, 1997; Zhang, Patuwo and Hu, 1998). This makes them able to handle more complex data than other typical statistical models.

Kaastra and Boyd (1996) highlight additional advantages of ANNs. The authors mention that ANNs are able to tolerate noise, faults and other chaotic elements in data and are also better at handling data that is heavy-tailed. Svozil, Kvasnicka and Pospichal (1997) similarly mention that ANN models are robust and tend to perform well even when faced with large amounts of noise in a dataset.

However, artificial neural network models also present some downsides. One major criticism of these models is their “black-box nature”(Kaastra and Boyd, 1996; Tu, 1996; Svozil, Kvasnicka and Pospichal, 1997; Zhang, Patuwo and Hu, 1998; Nygren, 2004). The process used by an ANN to compute output values cannot be decomposed in an explicable manner. As a result, it becomes difficult to establish and validate the network’s internal decision-making process (Zhang, Patuwo and Hu, 1998; Cao, Leggio and Schniederjans, 2005).

Although the flexible nature of ANNs can be beneficial, this feature could dangerously lead to overfitting (Kaastra and Boyd, 1996; Tu, 1996; Zhang, Patuwo and Hu, 1998; Nygren, 2004). Overfitting occurs when an ANN model learns too well from the training data and fails to adequately generalize the information acquired during training (Lawrence and Giles, 2000). This negatively affects the ANNs performance during testing as the network could potentially define spurious causalities and incorrect patterns or forecasts (Tu, 1996; Zhang, Patuwo and Hu, 1998).

A further limitation is the trial-and-error methodology required in building a neural network. Unfortunately, there is no formal theory or defined method to assist with constructing the appropriate network for solving a particular problem (Kaastra and Boyd, 1996; Svozil, Kvasnicka and Pospichal, 1997; Zhang, Patuwo and Hu, 1998). Furthermore, the computational time and intensity demanded by neural network development is generally high due to the large amounts of data utilised (Zhang, Patuwo and Hu, 1998; Nygren, 2004).

3.3 Artificial Neural Networks in Finance Research

Research around the use of ANNs in finance has been gradually growing since the early 1990s. A review of the literature for the period of 1971-1996 was conducted by Wong and Selvi (1998). This review revealed that there had been no research on the application of ANNs in finance published before 1990. Furthermore, the review showed that more research had been done in the fields of bankruptcy predictions and stock price performance predictions. Many of these research papers suggested that ANNs performed better than typical statistical methods, particularly time-series analysis, due to their ability to exemplify non-linear relationships. Alongside this, other areas such as bond ratings, interest rate predictions and IPO pricing had also been explored (Wong and Selvi, 1998).

The subsequent discussion explores some of the diverse applications of ANNs in the field of finance.

Kryzanowski, Galler and Wright (1993) investigate the use of ANNs in stock selection. The aim of their research was to investigate whether ANNs can differentiate between stocks that generate positive returns and stocks that generate negative returns. The authors used an ANN with an algorithm that can detect patterns to learn the relationship between a company's stock return and the company's historical firm-specific information as well as historical macroeconomic information. Thereafter, the trained ANN was used to predict the nature of a company's future stock return, i.e. the returns were predicted as positive, neutral or negative.

The authors made use of firm-specific financial information for 120 public companies over the period from 1984 to 1989 (Kryzanowski, Galler and Wright, 1993). The firm-specific input variables used are listed in the table below.

Table 3.2: Firm-specific financial ratios used by Kryzanowski, Galler and Wright (1993)

Category	**Ratio**
Profitability	Gross Margin Net Profit Margin Total Asset Turnover Fixed Asset Turnover Return on Total Assets Return on Equity
Debt	Debt Ratio Debt-to-Equity Ratio Interest Coverage Ratio Long-term Debt Ratio
Liquidity and Activity	Current Ratio Quick Ratio Accounts Receivable Turnover Accounts Payable Turnover

In addition to these firm-specific financial ratios, seven macroeconomic input variables were also used in the ANN model. Some examples of these include GDP, the 90-day Treasury bill rate, and the CPI (Kryzanowski, Galler and Wright, 1993:24).

The pattern-recognition algorithm used by Kryzanowski, Galler and Wright (1993) is formally known as the Boltzmann Machine (BM) and this classifies as a supervised learning method. During the training phase, the BM studies the relationship between input variables and the corresponding output. Furthermore, the BM gradually increases the weights of the inputs which are frequently present when a specific output occurs (Kryzanowski, Galler and Wright, 1993). Although the BM can function with continuous variables, it requires large amounts of data in order to learn and generalize accurately. Due to the relatively small dataset used, the authors manually coded the input variables into discrete, categorical values. The macroeconomic variables were coded as bad, average or good based on annual performance while the firm-specific financial ratios were coded as upward, stable or downward based on each ratio's respective trend (Kryzanowski, Galler and Wright, 1993:24).

Kryzanowski, Galler and Wright (1993) used the ANN for a two-state output classification, i.e. positive or negative returns, and also for a three-state output classification, i.e. positive, neutral and negative returns. The results of their tests showed that the ANN had a total accuracy of 66.4% when predicting stock returns in the two-state output classification and a total accuracy of 45.6% for the three-state output classification. The authors conclude that ANNs and in particular, the BM, have the capability for stock selection however, to achieve more accurate results, a larger amount of training data and a longer testing period would be beneficial (Kryzanowski, Galler and Wright, 1993:26).

A research paper by Burger (2011) explores the use of artificial neural networks in predicting the nature and the magnitudes of stock returns one year into the future. This research expands on the study by Kryzanowski, Galler and Wright (1993) to produce comparable results in the context of the South African stock market. Consequently, the study was conducted on a broad sample of 18 stocks listed on the Johannesburg Stock Exchange and over an investigation period extending back as early as 1991 (Burger, 2011).

The study made use of various multilayered, feed-forward ANN models since several hidden layer sizes and activation function specifications were explored. Additionally, the ANN models in the study were trained using the Broyden-Fletcher-Goldfarg-Shanno method for optimisation. The primary input factors used were three firm-specific financial ratios based on value-investing principles, namely, price-to-book ratio, earnings yield and debt-to-equity ratio. Alongside these variables, historic stock returns and selected historic economic indicators were added to some network models to enhance forecasting accuracy (Burger, 2011).

Similar to Kryzanowski, Galler and Wright (1993), two-state and three-state classifier ANNs were implemented to predict the nature of future returns. However, in contrast to Kryzanowski, Galler and Wright (1993), the three-state classifier ANNs categorized returns in the following three groups: lower than 10%, between 10% and 30% and lastly, greater than 30% (Burger, 2011). For predicting the magnitude of future returns, a mean absolute deviation (MAD) benchmark of 5% was used to evaluate network accuracy (Burger, 2011).

It is found that the inclusion of all input variables resulted in the best performing ANN. However, evidence from sensitivity analyses revealed that economic indicators provided the most significant information. The best performing ANN achieved an overall accuracy of 80.2% during two-state classification (Burger, 2011). This result surpassed that of Kryzanowski, Galler and Wright (1993). Moreover, the false positive and false negative rates were notably low at 12.9% and 6.9% respectively (Burger, 2011). In the case of three-state classification, the best performing ANN achieved an accuracy of 60.1% and likewise outperformed the results of Kryzanowski, Galler and Wright (1993) (Burger, 2011). For the quantitative predictions, performance across all implemented ANNs was substantially poor. The best trained ANN achieved an MAD of 20% during testing which far exceeded the 5% benchmark set (Burger, 2011). The author concludes that ANNs designed for the purpose of classification display encouraging prospects for future use in stock return prediction. However, regression-type ANNs designed to predict stock return magnitudes require significant improvement to allow for more accuracy in future applications (Burger, 2011).

Eakins and Stansell (2003) investigate the ability of neural networks to generate superior returns on a risk-adjusted basis using value-investing principles. The study was conducted on all Compustat-listed stocks with a market capitalization of 150 million US dollars. The investigation period spanned from 1975 to 1996.

A multilayer feed-forward neural network was used for the study. The input layer consisted of six firm-specific financial ratios which are often indicative of value. These include price-to-sales ratio, price-to-earnings ratio, dividend yield, price-to-book ratio, market capitalization and price-to-cash flow ratio. The hidden layer of the network contained 3 hidden neurons. The output layer comprised one neuron which represented the total return expressed as a percentage. Additionally, the backpropagation algorithm along with the hyperbolic tangent (tanh) activation function were used for training the network (Eakins and Stansell, 2003).

The results from the research showed evidence of the ability of neural networks to select outperforming portfolios of stocks using value-based financial information. The portfolios selected by the neural network were able to beat two major indices, the S&P 500 and the Dow Jones Industrials, as well as the original full sample of stocks. Furthermore, the authors demonstrate that the outperformance of the neural networks persists even on a risk-adjusted basis. In other words, the portfolios selected by the neural network were able to earn higher returns with a lower risk. The authors conclude “This paper provides additional support for the position that investing in value stocks provides superior risk-adjusted returns. We found that neural networks select portfolios that are generally superior alternative selection methods" (Eakins and Stansell, 2003:96).

In a study by Cao, Leggio and Schniederjans (2005), the use of artificial neural networks in predicting returns in the Chinese stock market is investigated. The research was performed on 367 stocks listed on the Shanghai Stock Exchange. The period of study was a 4-year period extending from January 1990 to December 2002, however, daily firm-specific data and stock returns were used.

The predictive accuracy of these non-linear ANN models is compared to that of more popular linear asset pricing models, namely, the CAPM and the Fama & French three-factor model (Cao, Leggio and Schniederjans, 2005). The authors make use two different neural network models which correspond to the two linear regression models to allow for a more direct comparison. In the first case, a univariate ANN was built with one input neuron in the input layer, a range of four to 10 hidden neurons in the hidden layer and finally, one output neuron in the output layer (Cao, Leggio and Schniederjans, 2005). The firm-specific input variable used was beta and this allowed the model to correspond to the CAPM. In the second case, a multivariate ANN model was built with three input neurons and a range of five to 15 hidden neurons. The structure of the output layer remained unchanged. The three input variables used were beta, market capitalization and the book-to-market ratio. The second ANN model corresponded to the Fama and French three-factor model. The backpropagation algorithm was used for training both neural network models (Cao, Leggio and Schniederjans, 2005).

Cao, Leggio and Schniederjans (2005) use three different error measures to assess the accuracy of each model: Mean Absolute Deviation, Mean Absolute Percentage Error and Mean Squared Error. It is found that the ANN models had better forecasting power than their linear counterparts across all the performance measures used. Additionally, an interesting discovery made was that the univariate linear and non-linear models exhibited superior predictive accuracy to their three-factor counterparts suggesting that the market's returns and consequently, a stock's beta, may be the only important variables for stock returns in the Chinese equity market (Cao, Leggio and Schniederjans, 2005).

Yoon and Swales (1991) compare the efficacy of artificial neural networks against that of multiple discriminant analysis in the field of stock price performance prediction. The study made use of two samples of data from two popular investment sources, namely the Fortune 500 and Business Week's Top 1000. The first dataset was extracted from the Fortune 500 list and was used for the training phase. This dataset comprised 58 companies from five industries with the highest yearly returns. The second sample of data, obtained from Business Week's Top 1000, was used for the testing phase and consisted of 40 companies from 10 highly-valued industries (Yoon and Swales, 1991).

The forecasts of stock price performance were based on qualitative information extracted from the letter to shareholders found in company annual reports. This information contained themes represented by nine variables. According to Yoon and Swales (1991), these variables include:

- Growth
- New products
- Confidence
- Strategic plans
- Anticipated gains
- Anticipated losses
- Short-term optimism
- Long-term optimism
- Economic factors beyond the company's control (Yoon and Swales, 1991:158)

The ANN model designed for the research was a feed-forward network with four layers. The input layer contained the nine firm-specific qualitative variables. There were two hidden layers in the model. The number of hidden neurons in each hidden layer were determined experimentally which resulted in four neurons in the first hidden layer and one neuron in the second. The output produced by the network was categorical in nature and had two classifications, namely, a poor-performing or a well-performing stock price. The ANN model made use of the logistic sigmoid activation function together with the backpropagation learning algorithm (Yoon and Swales, 1991).

Yoon and Swales (1991) find that the ANN model achieved more accurate stock price performance forecasts than the multiple discriminant analysis method, indicating that the non-linear model was a more suitable forecasting tool. Moreover, the authors also discovered that there is an optimal number of hidden neurons to include in the hidden layers of an ANN model which influences the predictive capability of the network. Increasing the number of hidden neurons beyond this optimal number weakens the performance of the ANN (Yoon and Swales, 1991:161). Yoon and Swales (1991) highlight that a limitation of ANNs, which stems from their black-box nature, is the difficulty in interpreting the relative significance or weight of each input variable due to the presence of hidden layers. Nonetheless, the authors still promote the application of ANN methods due to their superior performance (Yoon and Swales, 1991).

In a paper by Leshno and Spector (1996), the authors assess the predictive ability of a variety of ANN models specifically in bankruptcy. The models used differ in data set, network architecture and number of epochs. Furthermore, the authors compared the results from the neural network models to the results obtained from the typical discriminant analysis methods which are generally used for bankruptcy predictions (Leshno and Spector, 1996).

Leshno and Spector (1996) find that the neural network models perform much better when presented with larger amounts of data as this allows the networks to learn from a larger number of cases. The authors also find that more complex network architectures may lead to the problem of overfitting as the network would become too specified on the training data and would thereby fail to generalize the information presented to it. This usually results in the network losing its ability to make significantly accurate out-of-sample forecasts or predictions. Furthermore, the authors find that having a greater number of iterations does not always lead to an improvement of the networks forecasting ability (Leshno and Spector, 1996).

Multiple discriminant analysis and logistic regression analysis are the classical models that are used for bankruptcy predictions. However, when these models are compared to the neural network model, Leshno and Spector (1996) find that the neural network model provides more precise bankruptcy predictions.

Hamid and Iqbal (2004) make use of neural networks to forecast the volatility of the S&P 500 Index futures prices. The authors make use of a multilayered feed-forward neural network with an input layer, one hidden layer and an output layer (Hamid and Iqbal, 2004). Alongside this architecture, the backpropagation learning algorithm was used to train the network. The volatility forecasts from their model were then compared to implied volatilities extracted using the Barone-Adesi and Whaley option-pricing model as well as actual realized volatilities (Hamid and Iqbal, 2004).

Hamid and Iqbal (2004) find that the volatility forecasts from the neural network model are not significantly different from the volatilities which are realized in the market. It is also found that the results from neural network model outperform the implied volatility forecasts. The authors conclude that neural networks present promising applications in the field of finance. However, realizing this potential will require experimentation in order to successfully develop the appropriate models that will solve specific problems (Hamid and Iqbal, 2004).

3.4 Conclusion

This chapter has reviewed the limitations that multiple linear regression presents and has shed light on how these are overcome by artificial neural networks. Additionally, the general advantages and shortcomings of ANNs have been reviewed. This chapter has also presented research around the application of ANNs in the field of finance.

Artificial neural networks provide an alternative approach for modelling relationships between different variables in a dataset. These models are non-parametric by nature as they do not require prior assumptions about the data presented to them. ANNs are able to study the different relationships in a given dataset, generalize these relationships and accurately extrapolate on a similar unseen dataset. Furthermore, due to their non-linear structure, ANNs are able to capture non-linear relationships which would have otherwise been overlooked by typical linear modelling techniques.

Although artificial neural networks present advantageous modelling features, they are not free from weaknesses. The flexible nature of ANNs could lead to overfitting which negatively affects their out-of-sample performance. Furthermore, it is difficult to comprehensively determine the internal modelling process of an ANN. Additionally, designing an artificial neural network model is not straightforward as there are no formal theories or guidelines to follow.

Several researchers have investigated the use of ANNs in various areas in the field of finance. The overarching theme presented by the literature is the superiority of ANNs over other typical linear statistical analysis techniques. The findings from the different research papers and particularly from Kryzanowski, Galler and Wright (1993), Burger (2011), Eakins and Stansell (2003) and Cao, Leggio and Schniederjans (2005), present substantial motivation for the use of ANNs to explore the relationship between returns and firm-specific financial information.

4 Data

4.1 Introduction

This chapter reviews the data that is used in this research. The data consisted of 161 stocks sampled from a merged list of two stock indices. The indices used were the Russell 1000 Index, which consists of the largest 1000 public stocks in the United States, and the Standard and Poor's (S&P) International 700 index which comprises the 700 largest stocks outside the United States. The stocks in the sample dataset were listed in alphabetical order alongside their monthly corresponding firm-specific factors, prices and returns for the period from January 2001 to October 2014. This 166-month period constituted the investigation period for this research.

The sample dataset was originally obtained from the *Bloomberg* terminal located in the Chancellor Oppenheimer Library of the University of Cape Town. Professor Paul Van Rensburg, who is the supervisor of this research, kindly granted access to this dataset for use in this research. The dataset extracted from the *Bloomberg* terminal was transferred to *Microsoft Excel* which is compatible with *Statistica*, the statistical software package used in this research.

Possible biases that are present in the data and any necessary adjustments to the dataset are also considered in this chapter.

4.2 Data Description

As previously mentioned, the sample dataset is made up stocks from two indices, namely, the Russell 1000 Index and the S&P International 700 Index. The combination of the two indices was narrowed down to the 161 stocks which had relatively fuller sets of information necessary for this research. In view of the fact that the Russell 1000 Index is the larger index, the greater part of the stocks in the sample set are based in the U.S.

Majority of the stocks in the sample dataset are from the Information Technology, Industrial, Health Care and Consumer Discretionary sectors. Nonetheless, a total of ten sectors are represented in the sample.

Figure 4.1 below depicts the frequency distribution of the sample stocks by sector.

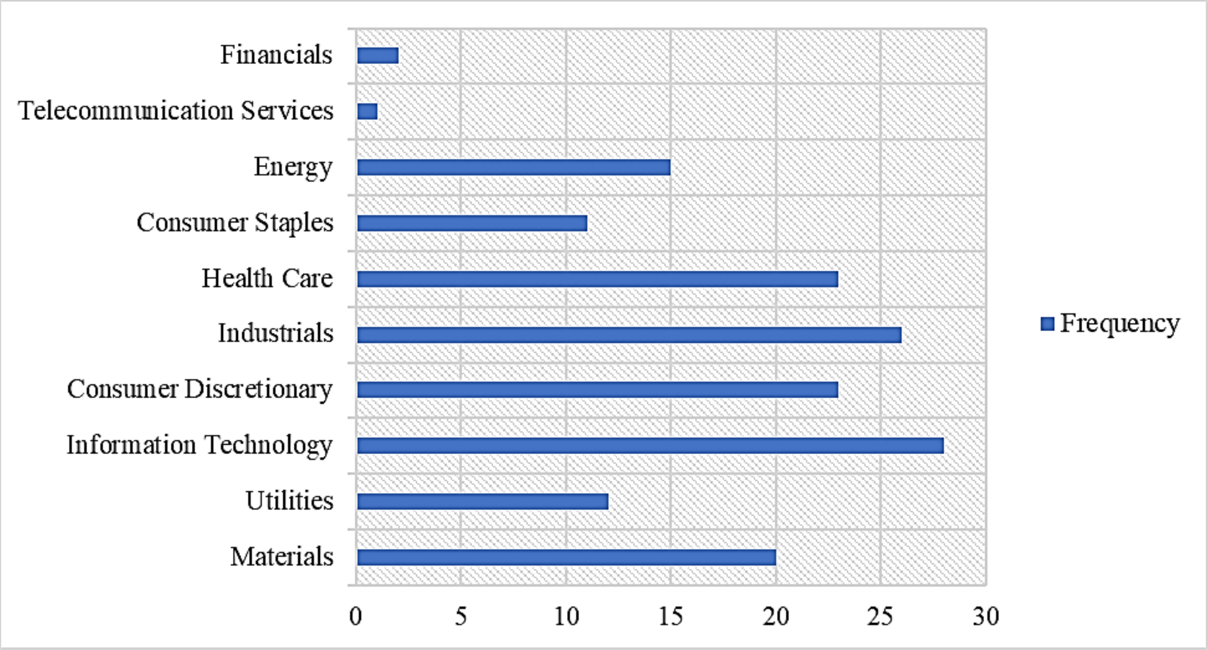


Figure 4.1: Sample Stocks Represented by Sector

4.3 Firm-Specific Factors

One of the aims of this research is to determine whether an artificial neural network which uses firm-specific factors as inputs can accurately predict stock returns and if so, which factors contribute most to the network's predictive ability. The use of firm-specific factors stems from the active investment management principle which suggests that an investor can predict future stock prices using fundamental analysis (Malkiel, 1999).

Prior research conducted by Kryzanowski, Galler and Wright (1993), Arnott, Hsu and Moore (2005), Ellis and Wilson (2005), Hodnett (2010), Ashwood (2014) and predominantly Hartzenburg (2015) was consulted to inform the selection of firm-specific factors that were included in the ANN models. The findings of these researches suggested that certain firm-specific accounting information has had significant relationships with stock returns in the past. The factors that were used as inputs in the ANN were extracted from company financial statements on the Bloomberg terminal for each stock in the sample set.

The firm-specific factors used in the ANN models can be classed into five style categories, namely, Size and Liquidity, Momentum, Value, Growth and Risk. Table 4.1 below depicts the different factors used, grouped according to their style categories with their corresponding standardised descriptors. According to Hartzenburg (2015), these firm-specific style factors were found to be significantly predictive of monthly returns on the global equity market.

Table 4.1: Firm-specific factors used in each ANN model (Hartzenburg, 2015)

Style Category	**Firm-specific Factor**	**Descriptor**
Size and Liquidity	Natural Log of Market Capitalisation Natural Log of Monthly Value Traded % Change in Monthly Volume: 12-Month	LNSIZE LNVOLUME VOLUME12
Momentum	One-month Prior Return Three-month Cumulative Prior Return Six-month Cumulative Prior Return	MOM1 MOM3 MOM6
Value	Cash Flow to Earnings Sales to Price Book Value to Price Payout Ratio	CFTOE SALESTP BVTP PR
Growth	Capital Expenditure Ratio % Change in Non-current Assets: 12-Month % Change in Sales: Six-Month Net Profit Margin Return on Equity % Change in Asset Turnover: Six Months % Change in EBITDA Margin: 12-Months	CAPEXR NCA12 SALESG PM ROE AT6 EBITDAM12
Risk	Trailing 12-Month Standard Deviation of Returns	TRL12STD

4.4 Potential Biases

Survivorship bias is the tendency to overlook 'non-surviving' firms in studies of performance (Brown et al., 1992). 'Non-surviving', in this case, usually refers to firms that have been subject to firm failure. The stocks used in this research are extracted from the Russell 1000 Index and the S&P International 700 Index. These two indices list the largest stocks by market capitalization in the U.S. and worldwide respectively. By virtue of this, the sample set suffers from survivorship bias.

Survivorship bias raises concerns for researchers and practitioners as it tends to create a degree of skewness in research findings, making them weaker (Brown et al., 1992). The aim of this research is to evaluate the performance of ANNs in predicting the returns of a sample of stocks, not to measure the actual performance of these stocks. Therefore, although the sample set suffers from survivorship bias, it should be noted that this is not a major concern in this research.

4.5 Necessary Adjustments to the Dataset

This research required a complete dataset without any incomplete fields to allow for an unconstrained and efficient performance by the ANN models. However, some firm-specific factors for specific stocks were not accessible for the entire period for which the sample dataset was extracted. Consequently, some adjustments had to be made. Authors such as Batista and Monard (2003) and Somasundaram and Nedunchezhian (2011) suggest replacing the missing fields of a variable with the median value for that variable in the sample dataset. Using this statistic avoids the introduction of either an upward or downward bias to the results of the analysis. For that reason, this proposed approach was used to replace any incomplete fields in the sample dataset. Furthermore, while the stocks present in the sample are listed in different countries, their constituent values were presented in one single currency, the US Dollar. This allowed for standardised results and simpler analysis.

4.6 Conclusion

This chapter has presented a description of the sample dataset used in this research. This sample comprises the top 161 stocks extracted from a combined and alphabetically ordered list of stocks from the Russell 1000 and S&P International 700 indices. The sample also consists of monthly price, return and firm-specific information for each stock over the period of January 2001 to October 2014. Most of the stocks in the sample are based in the United States. However, all sectors of the market are represented.

The input variables used for the ANN model have been identified based on similar research conducted by other authors. A possible bias has been considered and although the sample dataset suffers from survivorship bias, it has been determined that this is not a major concern for this study. Finally, the necessary adjustments to the sample dataset have been discussed. Particularly, incomplete fields have been replaced with median values to provide a complete dataset prior to the implementation of the ANN models.

5 Method

5.1 Introduction

This chapter discusses the methodology used to build and train the ANN models and to test the performance thereof for this research. This chapter also provides a description of the software used for the analysis.

The methodology applied is inspired by the work of Kaastra and Boyd (1996) who present and eight-step method for building an ANN model for time series forecasting in a finance and economics context.

Kaastra and Boyd (1996) present this eight-step method as follows:

- “Step 1: Variable selection
- Step 2: Data collection
- Step 3: Data pre-processing
- Step 4: Training, testing, and validation sets
- Step 5: Neural network paradigms
  - number of hidden layers
  - number of hidden neurons
  - number of output neurons
  - transfer functions
- Step 6: Evaluation criteria
- Step 7: Neural network training
  - number of training iterations
  - learning rate
- Step 8: Implementation” (Kaastra and Boyd, 1996:219)

This process may require several iterations (or epochs) in order to achieve an optimal model.

The first two steps, Variable Selection and Data Collection, have been discussed in the most recent chapter therefore this chapter will review and discuss the subsequent six steps of the method in the subsequent sections.

5.2 Data Pre-processing

To ensure that the ANN model functions efficiently, i.e. trains quickly and produces significantly accurate results, the data presented to the network requires pre-processing. As mentioned in the previous chapter, the sample dataset initially contained incomplete fields as some factors for the stocks were not retrievable for the entire period for which the data was retrieved. However, the sample was adjusted by replacing the missing fields of a variable with the median of that variable over the entire investigation period. This method prevented the network from learning from an incomplete dataset.

A further pre-processing procedure required for the data was the scaling of the input variables to values which lie between 0 and 1. This is because the input variables used in the ANN model have a range of different magnitudes as depicted by their respective distributional histograms found in Appendix A. For example, the natural log of monthly volume traded reaches values of 20 while a variable such as the payout ratio only reaches a maximum of 0.9 across the different stocks and over the entire period. This difference could potentially cause problems for the ANNs during its training phase as the network may struggle to converge to a minimised error because some weights will be updated faster than others. In addition to this, scaling the input variables allows for better processing through the activation function. The most widely used scaling techniques are the Min-Max scaling and Z-score normalisation methods (Kaastra and Boyd, 1996). These are expanded below.

- Min-Max Scaling: $X_{norm} = \dfrac{X - X_{min}}{X_{max} - X_{min}}$
- Z-score normalisation: $X_{norm} = \dfrac{X - \mu(X)}{\sigma(X)}$

The Min-Max Scaling method was used for the purposes of this study.[1] Given that the input variables to be used in the networks were scaled, the outputs produced by the networks had to be scaled back to an explicable variable, i.e. one-month forward returns.[2]

---

[1] This scaling process was executed automatically by *Statistica*.

[2] The reverse scaling process was also completed automatically by *Statistica*.

This was performed before any comparisons with the target or desired output were made for the error calculations.

5.3 Training and Testing Datasets

Before the ANN model was implemented, the sample dataset had to be divided into two subsets: the training set and the testing set. The training set is the larger set of data and was used to train the neural networks to observe the patterns present in the data and to update the weights of the input variables. The training set was also used to assess the performance of the ANNs, i.e. their ability to reach a global minimum error. Once the networks were adequately trained, the testing set was used as a final out-of-sample assessment of the best-trained network.

This research made use of a 70:30 split between the data. Since the full dataset to be used consists of 166 months of data for each stock, the 70:30 split resulted in a training period from January 2001 to December 2010 and a testing period from January 2011 to October 2014.

5.4 Neural Network Structure

The architecture of an ANN describes the general structure of the network, i.e. how many layers the network contains and how many neurons are present in each layer.

The simplest layer to determine is the input layer. There is typically only one input layer in which each input factor is represented by one neuron. Therefore, the number of neurons in the input layer will be equal to the number of input factors or independent variables being tested in the network. For this particular research, each stock has 18 firm-specific factors and therefore each ANN model designed had one input layer with 18 neurons.

Kaastra and Boyd (1996) note that theoretically, a neural network with one hidden layer and an adequate number of hidden neurons has the ability to model any continuous function well. Furthermore, the authors mention that in practice, there is evidence of neural networks with a maximum of two hidden layers also performing well. The presence of hidden layers in a neural network enhances the network's capacity to generalise patterns and hence enhances the network's performance. However, there is a danger of having too many hidden layers as this increases the network's processing time and increases the risk of the network overfitting the data (Kaastra and Boyd, 1996). For this study, one hidden layer has been used in each network's architecture.

There is no standard procedure for determining the optimal quantity of hidden neurons to include in the hidden layer of an ANN. However, Kaastra and Boyd (1996) draw attention to some of the heuristics developed and proposed by other researchers. These are discussed below:

- Masters (1993) proposes a geometric pyramid rule in which the number of neurons in the hidden layer of a three-layered network would range from $(0.5 \times \sqrt{n \times m})$ to $(2 \times \sqrt{n \times m})$, where n the number of neurons in the input layer and m is the number of output neurons.
- Bailey and Thompson (1990) recommend that the number of hidden neurons should be $0.75 \times n$ where n is the number of input neurons.
- Katz (1992) suggests that the number of hidden neurons should range between $(0.5 \times n)$ to $(3 \times n)$, where n is the number of input neurons.
- Although not a heuristic, research by Yoon and Swales (1991) suggests that there is an optimum number of hidden neurons beyond which, network performance diminishes.

Selecting the best number of hidden neurons to include in the hidden layer required some experimentation guided by the different methods proposed above. This involved designing a few neural network models each with a different number of hidden neurons in the hidden layer. The aim was to select the model which performs best with the lowest number of hidden neurons in order to minimise the risk of overfitting (Kaastra and Boyd, 1996). Based on the heuristics proposed above, the number of hidden neurons that were investigated were three, nine, 13, 18 and 30 hidden neurons.

The size of the output layer was relatively more straightforward to ascertain. In this research, the ANN was used to predict the one-month-forward returns (the dependent variable) based on information extracted from 18 firm-specific input factors (the explanatory variables). Consequently, only one output or dependent variable was required.

The ANN architecture required the addition of an activation function in both the hidden and output layers. This mathematical function is used to ascertain the output computed by a processing neuron. According to Kaastra and Boyd (1996), the sigmoid activation function is the most common function used for data involving time series as this function possesses valuable characteristics such as non-linearity and differentiability. For the purposes of this research, the hyperbolic tangent sigmoid function was selected as the activation function to be used in both the hidden and output layers. This function allows for the computation of output values which lie in the range of -1 to +1. This is advantageous as stock returns generally fall within this range. The mathematical form of the activation function to be used is as follows (Zhang, Patuwo and Hu, 1998):

$$f(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

5.5 Evaluation Criteria

This step of ANN design entailed defining the success criteria that will be used to assess the neural network's performance during both the training and testing phases.

For the training phase, an ideal error function, which the network aimed to minimise, had to be selected. Zhang, Patuwo and Hu (1998) draw attention to the various error functions that are commonly used in the literature to assess the performance of ANNs. These functions include:

- The sum of error squared (SSE) $= \sum(e_t)^2$
- The mean square error (MSE) $= \frac{\sum(e_t)^2}{N}$
- The root mean square error (RMSE) $= \sqrt{MSE}$
- The mean absolute deviation (MAD) $= \frac{\sum |e_t|}{N}$
- The mean absolute percentage error (MAPE) $= \frac{1}{N}\sum |\frac{e_t}{y_t}| \times 100$

where e_t is the prediction error at time t, y_t is the observed value at time t and N is the number of error observations (Zhang, Patuwo and Hu, 1998:51).

The sum of squared error function is one of the popular and frequently used functions in literature (Kaastra and Boyd, 1996; Zhang, Patuwo and Hu, 1998) and was employed for the training phase. This function measured the sum of the squared errors between a network's predicted returns and the corresponding actual returns observed in the training sample. Ideally, a well-trained ANN will minimise this function and will produce an SSE close to or equal to zero. This would suggest that there is little to no difference between the network's predicted returns and the actual returns in the training sample and that the network is a good fit.

Forecasting ability was measured by the information coefficient. This is essentially the correlation between a network's predicted returns and the actual returns in the sample. Generally speaking, a correlation of 0.1 between the predicted and actual returns is desirable (Grinold and Kahn, 2000) and any result less than 0.1 would suggest that a network's predictive ability is weak. This evaluation criterion was used to assess the performance of the different ANNs implemented during the training phase. This measure was also used during the out-of-sample testing phase to assess the performance of the best-trained ANN model.

Another functional method which was used to evaluate the best-trained ANN's out-of-sample performance was the degree of error between the network's forecasted returns and the actual returns during the testing period. This was measured using the mean absolute deviation error function which was mentioned in an earlier section. The MAD can be intuitively interpreted as the average prediction error (in absolute value or magnitude terms) that the ANN makes when forecasting returns. Although there is no formal research to establish an optimal MAD target, Burger (2011) suggests that a 5% MAD is desirable for a regression-like ANN.

To complement the aforementioned performance measures, a more visual illustration of the best-trained network's predictive ability was produced by means of a scatterplot and a line of best fit between the network's forecast returns and the actual realized returns. These plots were employed to illustrate the distribution of the ANN's predicted returns around the line of no prediction error. Essentially, if the network is performing effectively, the scatter points should lie relatively close to the line of zero prediction error and there should be little to no noise in the plot. In addition to this, the line of best fit should have a slope and intercept close to or equal to 1 and 0 respectively.

To determine which firm-specific factors contributed most to the predictive power of each ANN model, a global sensitivity analysis was performed. This analysis provided a ranking of the input variables by relative importance for each individual ANN model.

5.6 Neural Network Training

The basic idea behind neural network training is to facilitate the learning or pattern recognition process of the network. During this phase, the network aims to ascertain the optimal weights of each input (explanatory variable) required to produce the necessary output (dependent variable). The initial weights to be used by each network during its training phase were initialised randomly using a uniform distribution over the interval $[-1, 1]$.[3] Thereafter, the network learned from the data and updated these weights appropriately. As previously mentioned, the networks aimed to minimise the error function which measures the discrepancy between the networks' computed outputs and the target outputs during the training phase. In other words, the networks trained to find a global minimum for the error function (Kaastra and Boyd, 1996).

Kaastra and Boyd (1996) draw attention to two different approaches pertaining to the optimal stoppage of the training phase for the network. The first approach proposes that the training should only be stopped once there is no improvement in the minimisation of the error function. At this point, the network's error would have converged to its global minimum. The second approach suggests that there should be a pre-specified number of iterations for which training occurs. Thereafter, the network's performance should be assessed, i.e. the error should be evaluated, and then training should be resumed if necessary. Nygren (2004) describes these two methods as late and early stopping respectively.

The second approach, early stopping, allows for shorter training periods and reduces the risk of overfitting. However, it is criticised because it does not provide any information on whether additional training would improve the network's performance and therefore the researcher may not know the best time to stop training (Kaastra and Boyd, 1996; Nygren, 2004) On the other hand, late stopping allows the network to reach a global minimum error but it is criticised because it does not necessarily provide assurance that a global minimum will indeed be reached (Kaastra and Boyd, 1996).

Some studies show that convergence to a global minimum can be achieved with around 85 to 5000 training iterations (Deboeck, 1994; Klaussen and Uhrig, 1994). Nonetheless, there is evidence of the use of higher training iterations. For example, Ashwood (2014) makes use of a maximum number of epochs or iterations of 100000, however, these network iterations were performed with a supercomputer.

---

[3] This configuration was executed by *Statistica.*

The number of training iterations used is also dependent on the network's learning rate. These two elements are inversely related – a network with a relatively lower learning rate requires a higher number of iterations in its training phase. The learning rate typically ranges between 0.1 and 0.9 (Kaastra and Boyd, 1996). If the learning rate is too low, the network may take too long to train and converge to a global minimum error. On the other hand, a significantly high learning rate may cause the network to generalize inadequately and therefore perform inefficiently.

For the purposes of this study, the network's training process was designed with a learning rate of 0.1 and 200 iterations. This was motivated by the need to prevent the network from learning too quickly and overfitting the data as well as the need to ensure that the network converges to a global minimum error. A total of 161 stocks were used for this research and these training specifications offered a tolerable computational time.

5.7 Implementation and Software

A summary of the network specifications that were held constant during the training and testing phases is presented in the table below. The network parameter that was varied during the training phase was the number of hidden neurons in the hidden layer.

Table 5.1: Network specifications held constant during training and testing

Parameter	**Setting**
Network Architecture	One input layer, one hidden layer, one output layer
Input layer size	18 firm-specific factors
Output layer size	One neuron representing one-month forward returns
Activation Function	Hyperbolic tangent sigmoid function
Data scaling	Min-max scaling
Weight initialisation	Randomly by Uniform distribution over $[-1, 1]$
Training period	10 years (January 2001 – December 2010)
Out-of-sample testing period	3 years, 10 months (January 2011 - October 2014)
Learning algorithm	Backpropagation with gradient descent learning
Error Function	Sum of Square Errors (SSE)
Learning rate	0.1
Number of epochs	200

As briefly mentioned in Chapter 3 and throughout this chapter, the software used to construct, train and test the ANN models used in this study was *Statistica*. This software contains a comprehensive range of statistical analysis, data mining and particularly automated neural network functions which assisted conveniently for this research.

The neural network function in the *Statistica* software allows for straightforward configurations for different neural network models. Alongside the ability to manually specify the input and output variables to be used, the function also allows for user-modified specifications for training and testing samples, hidden layers, input weight initialization, activation functions, training algorithms, learning rates and epochs.

One valuable feature of using the *Statistica* software is its rapid computational time. Given the large scale of data and the computational intensity required to train and test the different ANN models, a software with a quick response time serves as a useful solution. Another advantage is the software's ability to automatically execute the scaling of input and output variables which is essential for ANN modelling and analysis as discussed in Section 5.2 above.

5.8 Conclusion

This chapter has provided a detailed discussion of the methodology applied to conduct the analysis for this research and a description of the software used.

The Kaastra and Boyd (1996) eight-step method for ANN design was comprehensively adopted to design the ANN models for this study. The fixed framework for the network models was a multilayered feed-forward architecture with one input layer containing 18 input neurons, one hidden layer with a varying number of hidden neurons and one output layer with one neuron.

The network was configured and implemented with a hyperbolic tangent sigmoid activation function and a learning rate of 0.1 over 200 iterations. The network training process was initialised by the min-max scaling of the input variables which was automatically executed by *Statistica*, the software of choice for this study. The weights for the input variables were generated randomly using the Uniform distribution over the interval $[-1, 1]$. Training occurred over 120 months and each network's performance was evaluated using the SSE function and the information coefficient.

During the testing phase, the best-trained network's predictive power was measured by the MAD error as well as the information coefficient between the predicted returns and the actual returns. A global sensitivity analysis was applied to reveal which firm-specific factors were most important for each network's predictive performance.

6 Research Findings and Analysis

6.1 Introduction

This chapter assesses the results and findings obtained from the application of the methodology explained in the previous chapter. The results presented will attempt to contribute to the body of findings around the use of ANNs in finance. In particular, the results will aim to add to research about the use of ANNs in directly predicting stock returns from various firm-specific factors which have been demonstrated to have considerable effects on the cross-sectional returns in the global equity market.

Section 6.2 presents the results from the networks' training phases while Section 6.3 illustrates the findings from the out-of-sample testing phases. The results of the global sensitivity analysis done to determine the most significant firm-specific factors are presented in Section 6.4. The chapter's conclusion is presented in Section 6.5.

6.2 Network Training Analysis

The training phase is crucial for a neural network's out-of-sample performance. This phase allows a network to minimise its predictive error as it learns and generalizes patterns between input and output variables. For this study, five different networks were trained, each with a different hidden layer architecture. The SSE function and the information coefficient were used to assess the training performance of each network. The SSE function measured the sum of the squared errors between each network's forecasted returns and the corresponding observed returns in the training sample. The information coefficient measured the correlation between each networks' forecasted returns and the actual returns. As mentioned earlier, an SSE outcome close to or equal to zero and an information coefficient of 0.1 are desirable targets (Grinold and Kahn, 2000).

The table below shows the minimum SSE results that each network achievde during training.

Table 6.1: Training error for the different configurations of ANN models

	Network	**Training SSE**
1	MLP 18-3-1	0.006018
2	MLP 18-9-1	0.006061
3	MLP 18-13-1	0.006075
4	MLP 18-18-1	0.006142
5	MLP 18-30-1	0.006206

From the table above, it is clear that the network which achieved the smallest SSE during training was the network with the smallest number of hidden neurons, MLP 18-3-1. However, all the neural network models trained well as each minimised SSE is close to zero. These results are also illustrated by the training error graphs of each network found in Appendix B. The average SSE achieved across all five networks during training was 0.0061004.

Further examination of table 6.1 above indicates that the training error increased as the number of hidden neurons increased. This, however, does not imply that the network with the smallest training error performed the best during training. This is demonstrated in table 6.2 below which shows the correlation between predicted and actual returns achieved by each network during training.

Table 6.2: Training information coefficient for the different configurations of ANN models

	Network	**Training Information Coefficient**
1	MLP 18-3-1	0.109065
2	MLP 18-9-1	0.109142
3	MLP 18-13-1	0.110316
4	MLP 18-18-1	0.108954
5	MLP 18-30-1	0.107756

Each network performed quite well during training, achieving the desirable information coefficient of 0.1 or more between the networks' predicted returns and the actual returns from the sample. Nonetheless, the third network, with 13 neurons in its hidden layer, was the best performing network.

Table 6.1 suggests that a downward trend exists in the performance of a network as the number of neurons in the hidden layer increases. This would imply that the network with the least neurons in its hidden layer achieved the best training results. However, table 6.2 suggests that network performance improves as the number of neurons in the hidden layer increases until a particular point, after which, network performance deteriorates. In other words, there appears to be an ideal maximum number of hidden neurons. This finding supports the prior results of Yoon and Swales (1991) who note that increasing the number of hidden neurons past the optimal point does not enhance network performance but rather, worsens it.

The conflicting results presented by these two tables confirm the prior observation regarding the lack of a standard procedure to determine the ideal number of neurons to include in the hidden layer. Although there are established heuristics in place, it is evident that the optimal quantity of hidden neurons depends on other factors. Consequently, experimentation is essential for the process.

For this study, network three, MLP 18-13-1, was selected as the best-trained network to use as the primary network for the out-of-sample testing phase. This network achieved the best information coefficient during training. Furthermore, this network's training error, although not the lowest, was below the average. The full depiction of this network's structure and its connection weights can be found in Appendix C.

6.3 Out-of-Sample Testing Analysis

The generalised predictive ability of the best-trained ANN was evaluated using an out-of-sample testing dataset. This allowed the network to forecast returns for a new sample of data based on the information and patterns observed during the training phase. These predicted returns were then compared to the actual returns in the testing dataset. This testing process was also performed with the other trained ANNs. This was done to provide a basis of comparison for the best-trained ANN.

The MAD function and the information coefficient were employed as performance measures to assess each network's out-of-sample forecasting ability on the testing dataset. The MAD measured the average magnitude or absolute value of the errors made by the neural networks during out-of-sample forecasting. As previously discussed, a MAD of 5% and an information coefficient of 0.1 are desirable benchmarks (Grinold and Kahn, 2000; Burger, 2011). The table below depicts the MADs and information coefficients achieved by each ANN during the out-of-sample testing phase.

Table 6.3: MAD and information coefficient achieved during out-of-sample testing by each network

	Network	**MAD**	**Testing Information Coefficient**
1	MLP 18-3-1	6.82%	0.043
2	MLP 18-9-1	6.85%	0.048
3	**MLP 18-13-1**	**6.99%**	**0.054**
4	MLP 18-18-1	6.86%	0.053
5	MLP 18-30-1	6.86%	0.055

Based on the MAD criterion suggested in comparable research by Burger (2011), each network evidently underperformed the 5% benchmark. The average MAD achieved across all networks was 6.87% while the maximum MAD, achieved by network 3, was approximately 7%. Although the preferred network underperformed the benchmark, the degree of underperformance was not extreme as in the case of Burger (2011) who reported a 15% devaition from the MAD target. Furthermore, each network underperformed the benchmark for the information coefficient requirement.

In comparison to the preferred network, MLP 18-13-1, network 5 with 30 hidden neurons, achieved a higher information coefficient and a lower MAD. At first glance, it may seem that this network had a better out-of-sample performance. However, it is important to consider the risk of overfitting. Based on the network's larger hidden layer and poorer training performance, it is likely that the out-of-sample performance of network 5 may be at risk of overfitting. Therefore, for conservative reasons, network 3 remains the preferred network for assessing overall performance.

A visual representation of the preferred network's predictive ability is illustrated in the scatterplot below. This figure depicts the network's forecasted returns against the actual returns observed in the testing dataset while also showing the distribution of the forecasted returns about the line of no prediction error. As previously mentioned, all the predicted returns should ideally lie on or relatively close to this line, essentially exhibiting little to no prediction error. Additionally, the line of best fit should have a slope close to 1 and an intercept close to 0.

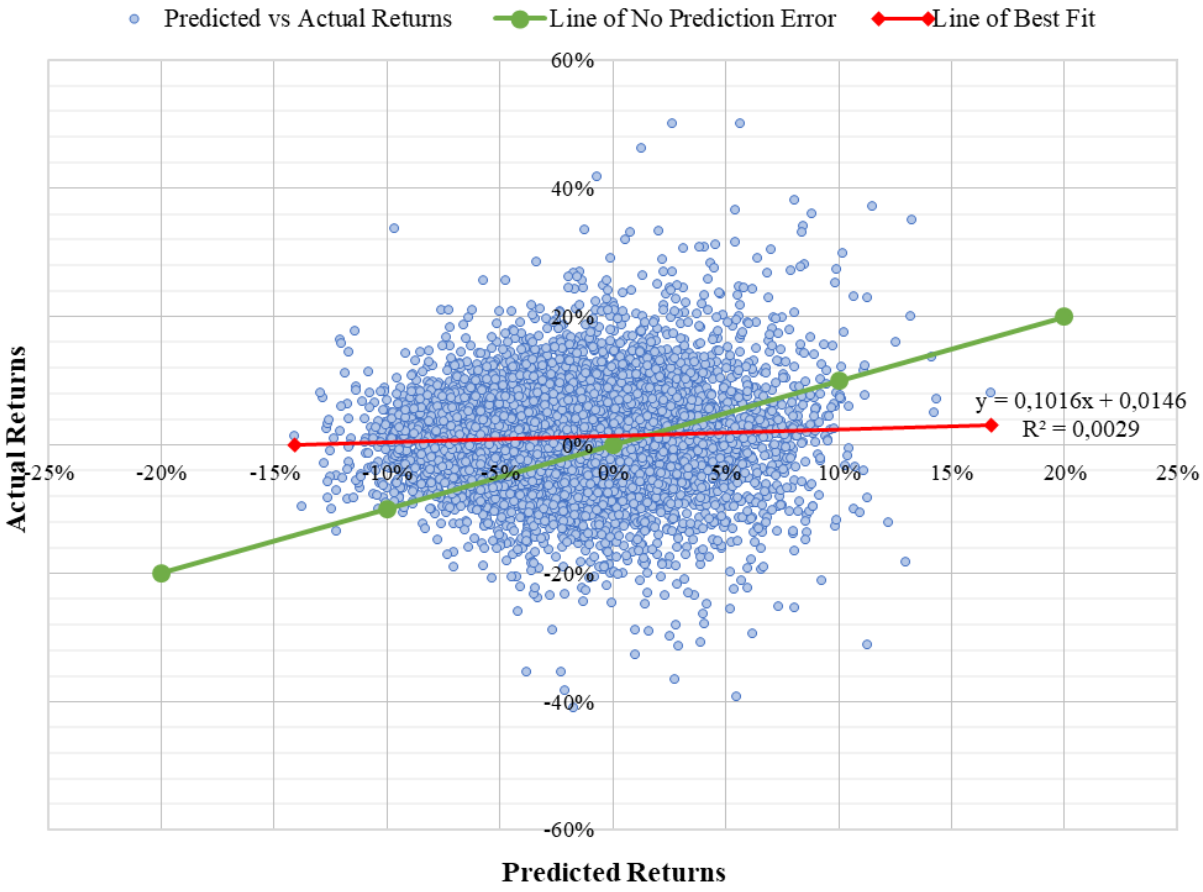


Figure 6.1: Predicted vs Actual Returns for MLP 18-13-1

From the figure above, it is observable that there is a considerable amount of noise present around the line of no prediction error. Additionally, there are notable outlying observations where the network either over or under-estimated returns significantly. The line of best fit suggest that there is a degree of bias between the network's predicted returns and the actual returns as the slope and intercept terms of this line are not equal to 1 and 0 respectively. Moreover, this bias changes from a negative to a positive bias as returns increase. The R^2 statistic indicates that there is a very weak relationship between the network's predicted returns and the actual returns in the testing sample.[4] Nonetheless, this visual illustration provides justification for the 7% MAD and the information coefficient of 0.054 achieved by the preferred network.

---

[4] The full regression output can be found in Appendix D.

For further analysis of the preferred ANN's out-of-sample performance, a histogram of its residuals was plotted. To briefly explain, a residual, in statistical terms, is generally defined as the difference between the actual and the predicted value. In this case, the residual is the difference between the actual returns in the dataset and the corresponding predicted returns produced by the network. The plot below depicts the distribution of the network's residuals produced during out-of-sample testing.

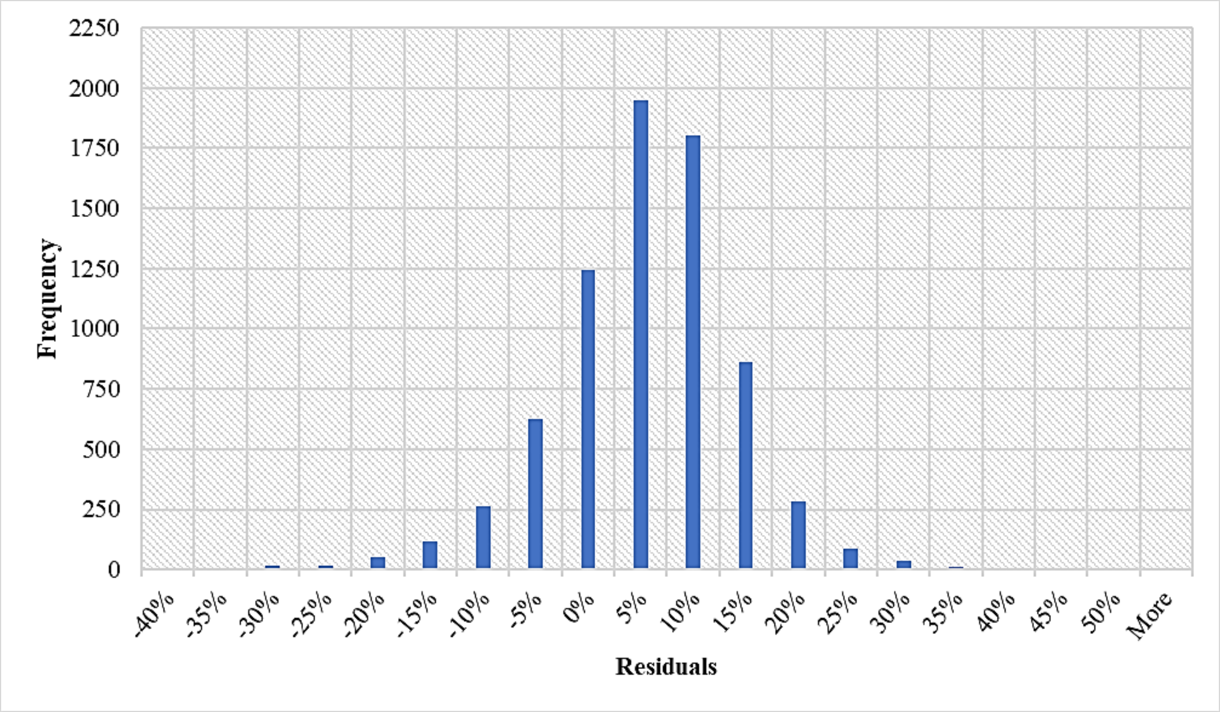


Figure 6.2: Histogram of Residuals for MLP 18-13-1

From the figure above, it is apparent that the majority of residuals from the network fell in the 0% to 5% range. This result outperforms that of Burger (2011) who reported residuals between 0% to 20% as the majority. Given the definition of a residual, it should be noted that any positive residuals, i.e. residuals greater than zero, imply that the network's predictions underestimate the actual returns. Consequently, there is evidence that the network is more likely to underestimate stock returns. This demonstrates the conservative nature of the preferred network. In contrast, Burger (2011) reported an equally likely chance of over or under-estimation by the ANN model.

6.4 Global Sensitivity Analysis

To determine the relative importance of the firm-specific input variables used in the ANN modelling process, a global sensitivity analysis was performed. This analysis investigated how the neural network's predictions would change, i.e. either increase or decrease, if each firm-specific input variable was to be altered. This process was performed individually for each of the five ANNs designed. Therefore, the results were produced in the context of each individual neural network model. The table below presents the firm-specific input factors in descending order according to their relative importance for each ANN model.

Table 6.4: Ranked relative importance of the firm-specific input variables used in each ANN model

	MLP 18-3-1	**MLP 18-9-1**	**MLP 18-13-1**	**MLP 18-18-1**	**MLP 18-30-1**
1	CAPEXR	CAPEXR	CAPEXR	CAPEXR	CAPEXR
2	ROE	VOLUME12	ROE	ROE	MOM3
3	VOLUME12	ROE	VOLUME12	MOM3	ROE
4	MOM3	MOM3	MOM3	VOLUME12	VOLUME12
5	SALESG	CFTOE	TRL12STD	CFTOE	CFTOE
6	AT6	EBITDAM12	CFTOE	SALESG	EBITDAM12
7	EBITDAM12	MOM1	NCA12	AT6	TRL12STD
8	NCA12	SALESG	AT6	EBITDAM12	MOM1
9	CFTOE	AT6	EBITDAM12	MOM1	SALESG
10	MOM1	NCA12	MOM1	NCA12	LNVOLUME
11	SALESTP	PR	SALESG	TRL12STD	AT6
12	PM	TRL12STD	PM	LNVOLUME	NCA12
13	PR	PM	PR	PM	PR
14	MOM6	MOM6	MOM6	MOM6	MOM6
15	TRL12STD	LNVOLUME	LNVOLUME	PR	PM
16	LNVOLUME	SALESTP	SALESTP	SALESTP	BVTP
17	BVTP	BVTP	BVTP	BVTP	SALESTP
18	LNSIZE	LNSIZE	LNSIZE	LNSIZE	LNSIZE

Based on the top 10 variables of each ANN model, it is evident that growth effects were most persistent. Particularly, the capital expenditure ratio (CAPEXR), return on equity (ROE), sales growth (SALESG), 12-month percentage change in non-current assets (NCA12) and six-month percentage change in asset turnover (AT6) were the most prevalent factors. Significant momentum effects represented by the three-month cumulative prior return (MOM3) and one-month prior return (MOM1) were also present. A noteworthy liquidity effect, 12-month percentage change in monthly volume traded (VOLUME12), was also dominant.

Size and value effects proved to be less pervasive. The natural log of market capitalisation (LNSIZE) was the least important firm-specific input variable across all the ANN models. The risk effect, characterized by the trailing 12-month standard deviation of returns (TRL12STD), produced mixed results across the different networks. However, this effect was particularly significant in the preferred network, MLP 18-13-1.

The results from the global sensitivity analysis are noteworthy especially given the characteristics of the stocks in the sample of data used for the study. Majority of the stocks in the sample fall into sectors which are generally classified as growth sectors. A prime example of this is the information technology sector. It appears that the ANN models designed all suggest that growth style factors are the most important variables for predicting the returns of a sample containing mostly growth stocks.The size style effect proved to be an insignificant factor across all the ANN models. It is difficult to explain the reason for this due to the black-box nature of the ANN models. However, the reason may be linked to the fact that all the stocks in the sample dataset used are large-cap stocks with very similar market capitalisations. This can be deduced from the distribution of the LNSIZE variable across all observations in the sample dataset which can be found in Appendix A.

6.5 Conclusion

This chapter has comprehensively presented and discussed the findings of this research. The results produced during network training revealed that all the five ANN models performed considerably well, achieving sum of square errors close to 0. Furthermore, each network succeeded in attaining the desired information coefficient of 0.1. The ANN model which performed the best during training was the ANN with 13 neurons in its hidden layer. This ANN model achieved the highest information coefficient and was therefore selected as the preferred network for further analysis.

During out-of-sample testing, all network models failed to attaining the target MAD of 5%, however, the deviations from this benchmark were moderate. The preferred network model achieved a slightly higher MAD as well as a slightly lower information coefficient than some of its peers. Nonetheless, this network remained the preferred network for further overall performance analysis in order to minimise the risk of reporting overfitted results. Additional regression analysis of the preferred network's predicted returns versus the actual returns in the testing sample revealed that the network performed poorly. Majority of the network's predicted returns fell within 5% of the corresponding actual returns in the sample. MLP 18-13-1 was also significantly conservative as the network has a tendency to underestimate returns in most instances.

Global sensitivity analysis revealed that size and value effects were least important in the predictive power of each ANN model. However, growth and momentum effects as well as one liquidity effect were especially pervasive. The risk effect produced mixed results across the different ANN models but was notably important in the preferred network.

7 Conclusion

7.1 Introduction

This chapter provides an overview of the findings of this research and the implications thereof on the objectives of this research. Limitations of the study are also discussed and recommendations for future research are presented. Finally, this chapter rounds up the research with concluding remarks.

The aim of this research was to investigate the use of artificial neural networks to predict stock returns using the firm-specific factors of several large stocks from various markets around the world. This objective gave rise to the following research questions:

- *Can artificial neural networks that employ firm-specific inputs accurately predict future stock returns?*
- *What hidden layer size leads to an optimal network forecasting performance?*
- *Which firm-specific inputs provide the most significant information for the return predictions made by the artificial neural networks in this study?*

This research attempted to answer these questions by firstly reviewing the theoretical background and literature around artificial neural networks and their use in the field of finance. Subsequently, several ANN models were designed and implemented on a training sample of data. The results of this then pointed to an optimal ANN model which was subsequently used for testing on an out-of-sample dataset. Several performance measures were employed to comprehensively assess the performance of the preferred neural network in order to present accurate conclusions.

This research was conducted on a sample of 161 stocks from the Russell 1000 and the S&P International 700 stock indices. The investigation period spanned a 166-month period from January 2001 to October 2014 with a 70:30 split for training and testing subsamples respectively. Prior research around the influence of style effects on the cross-section of global equity returns was used to inform the selection of the 18 firm-specific factors used as input variables for the ANN models.

Section 7.2 summarizes the research findings and the associated deductions intended for the research objectives. This section also discusses the link between the research findings and related financial theory. Section 7.3 presents the limitations encountered while conducting this research. Recommendations for future research are discussed in Section 7.4 and final remarks are presented in Section 7.5.

7.2 Summary of Findings

To evaluate the predictive ability of artificial neural networks, five multilayered feedforward networks were designed, each varying in hidden layer size. This network parameter was guided by heuristics presented in prior literature and was tested across the different network models. The findings suggest that network performance improves as the number of hidden neurons in the hidden layer increases until a specific point, after which network performance declines. This result supports prior research which suggests that there is an ideal quantity of neurons for the hidden layer(s) of a neural network. This parameter largely depends on the number of input and output variables in the neural network model. However, network performance is generally better with smaller hidden layers since the risk of overfitting increases as the size of the hidden layer increases.

Eighteen firm-specific factors from five broad categories namely, size and liquidity, momentum, value, growth and risk, were used as the input variables of the ANN to predict one-month forward returns of all the stocks in the sample. These firm-specific factors were selected based on prior empirical research by Kryzanowski, Galler and Wright (1993); Arnott, Hsu and Moore (2005); Ellis and Wilson (2005); Hodnett (2010); Ashwood (2014) and primarily Hartzenburg (2015). The networks were trained over the period from January 2001 to December 2010 and were tested over the period from January 2011 to October 2014. All five networks trained substantially well, with each achieving a minimum training sum of square error significantly close to 0 indicating that the networks fit the data well. Alongside this, each network model achieved the desirable information coefficient of 0.1 (Grinold and Kahn, 2000) between the forecasted and actual returns during training.

The network model with 13 hidden neurons in its hidden layer was prudently selected as the preferred model for out-of-sample testing as it trained relatively better than the other network models. During the testing phase, this model achieved an average prediction error, in absolute value terms, of approximately 7% and an information coefficient of roughly 0.05. A benchmark MAD of 5% was set based on research by Burger (2011) and similar to the training phase, a target information coefficient of 0.1 was set based on Grinold and Kahn (2000). Evidently, the results from the out-of-sample testing phase underperformed the desired targets. The visual illustration of these results depicted a significant amount of noise around the line of no prediction error. Furthermore, the regression between the model's predicted returns and the actual returns in the testing sample indicated a weak relationship and a significant bias. The conservative nature of this model was confirmed through its residuals which fell in the range of 0% - 5%. These results depicted that the network model is more likely to underestimate rather than overestimate stock returns.

A global sensitivity analysis was conducted to investigate the relative importance of each firm-specific factor used for the ANN models. The findings suggest that the growth style effects, particularly, the capital expenditure ratio, return on equity, sales growth, 12-month percentage change in non-current assets and six-month percentage change in asset turnover were the most persistent factors across all the ANN models. Three-month cumulative prior return, one-month prior return and 12-month percentage change in monthly volume traded were also significantly important factors. An interesting outcome of this analysis was the lesser pervasiveness of size and value style factors. Given that most of the stocks in the sample dataset came from sectors which are generally considered growth sectors, the results of the global sensitivity analysis illustrated that the ANN models suggested growth style factors as the most important variables for predicting returns. Furthermore, contrary to prior research and empirical findings (Hartzenburg, 2015), the ANN models classified the size effect as an immaterial factor, however, the reason for this is not easily determinable due to the black-box nature of the ANN models.

Broadly speaking, the results and findings of this study provide some, although relatively weak, evidence of return predictability. This may suggest a violation of the Efficient Market Hypothesis and may also present an argument against passive investment management. However, it is important to note that the validity of these arguments may be biased by data snooping since various model specifications were tested multiple times on the dataset to obtain the best-performing model with the lowest predictive error.

7.3 Limitations of the Research

Although the results from this study have provided some insights into the research problems discussed earlier, the research is not without limitations.

A limitation of this research is the sample of stocks used. As previously mentioned, the sample consisted of stocks from the Russell 1000 and the S&P International 700 indices. These two indices list the largest stocks by market capitalisation in the United States and the rest of the world respectively. By virtue of this, the sample suffered from survivorship bias as only surviving stocks listed over the entire investigation period were used for the study. While survivorship bias generally makes results weaker, it was not a huge concern for this research as the aim was to assess the forecasting performance of ANNs and not the actual performance of the stocks akin to a typical performance study. Nonetheless, it would be useful to also assess the predictive ability of ANNs using non-surviving or delisted firms. This is, however, beyond the scope of this research.

The sample of data used in this study initially contained incomplete entries as some firm-specific factors for certain stocks were only retrievable for a portion of the entire investigation period. To handle this issue, the missing fields of a factor were replaced with the median of that factor over the full investigation period. This adjustment may have negatively affected the performance of the ANNs particularly in the out-of-sample testing phase. This limitation, however, is not easily avoidable as some companies do not disclose all performance metrics on a monthly basis.

A further limitation of this study is the black-box nature of artificial neural networks. This attribute of ANNs restricts the extent to which results can be interpreted. In particular, it is difficult to clearly define the ANNs' proposed relationships between the firm-specific inputs and the returns of a stock. Moreover, the weights or coefficients of the firm-specific inputs are not easily translated as in the case of the typical multiple linear regression model. This limitation is unavoidable given the nature of ANN models.

7.4 Recommendations for Future Research

Future research into the use of artificial neural networks for stock return predictions should consider incorporating market returns as an explanatory or input variable. It would be useful to study whether the inclusion of this variable would improve the predictive performance of ANNs. It would also be interesting to observe whether ANNs identify the market as an important factor for predicting individual stock returns as suggested by CAPM and other linear asset pricing models which use the market as a risk factor. Additionally, although 18 firm-specific factors were used, this list is not exhaustive. Future research could explore more firm-specific factors for use as input variables in ANNs.

In this research, the size effect was the least significant variable across all the ANN models designed. Although the reasons for this are not easily explainable, there could be a link to the fact that only large-cap stocks were used in the study. It would be worth investigating whether this sensitivity analysis result persists with a more diverse sample of stocks (in terms of market capitalisation). Furthermore, in this study, the ANN models identified growth style effects as the most important variables and this could be due to the high concentration of growth stocks in the sample. Future research should investigate whether value style effects become more significant when a more value-based sample is used.

There are various other configurations of artificial neural networks that should be explored for the specific task of stock return prediction. A possible extension of this research is to compare the predictive performance of ANN models which significantly differ in architecture, activation function and training algorithm. This could help shed more light on the network configuration which performs best for stock return predictions.

7.5 Concluding Remarks

This research has explored whether artificial neural networks can be used to predict the stock returns of stocks in different markets across the globe using various firm-specific factors. This study has also examined which hidden layer configuration leads to the best network predictive performance. Additionally, the firm-specific factors which predominantly influenced the stock return predictions made by the ANNs have been identified. The results of this study have added to the body of research particularly around the use of artificial neural networks in stock return prediction. Yet, having underperformed the benchmarks suggested in prior research and existing practice, these results were unsuccessful in providing strong evidence in favour of the accurate predictive capability of artificial neural networks.

References

Arnott, R. D., Hsu, J. and Moore, P. (2005), 'Fundamental indexation', *Financial Analysts Journal* **61**(2), 83–99.

Ashwood, A. J. (2014), Portfolio selection using artificial intelligence, PhD thesis, Queensland University of Technology.

Bailey, D. L. and Thompson, D. (1990), 'Developing neural-network applications', *AI expert* **5**(9), 34–41.

Batista, G. E. and Monard, M. C. (2003), 'An analysis of four missing data treatment methods for supervised learning', *Applied artificial intelligence* **17**(5-6), 519–533.

Brown, S. J., Goetzmann, W., Ibbotson, R. G. and Ross, S. A. (1992), 'Survivorship bias in performance studies', *The Review of Financial Studies* **5**(4), 553–580.

Burger, E. (2011), Prediction of south african equity returns using artificial neural networks based on value-based investment principles, Master's thesis, Graduate School of Business - University of Cape Town.

Buscema, M. (1998), 'Back propagation neural networks', *Substance use & misuse* **33**(2), 233–270.

Cao, Q., Leggio, K. B. and Schniederjans, M. J. (2005), 'A comparison between fama and french's model and artificial neural networks in predicting the chinese stock market', *Computers & Operations Research* **32**(10), 2499–2512.

Deboeck, G. (1994), *Trading on the edge: neural, genetic, and fuzzy systems for chaotic financial markets*, Vol. 39, John Wiley & Sons.

Eakins, S. G. and Stansell, S. R. (2003), 'Can value-based stock selection criteria yield superior risk-adjusted returns: an application of neural networks', *International Review of Financial Analysis* **12**(1), 83–97.

Ellis, C. and Wilson, P. (2005), 'Can a neural network property portfolio selection process outperform the property market?', *Journal of Real Estate Portfolio Management* **11**(2), 105–121.

Fadlalla, A. and Lin, C.-H. (2001), ‘An analysis of the applications of neural networks in finance’, *Interfaces* **31**(4), 112–122.

Fama, E. F. (1970), ‘Efficient capital markets: A review of theory and empirical work’, *The Journal of Finance* **25**(2), 383–417.

Fama, E. F. (1991), ‘Efficient capital markets: II’, *The Journal of Finance* **46**(5), 1575–1617.

Fama, E. F. and French, K. R. (2010), ‘Luck versus skill in the cross-section of mutual fund returns’, *The Journal of Finance* **65**(5), 1915–1947.

Grinold, R. C. and Kahn, R. N. (2000), *Active portfolio management*, McGraw Hill New York, NY.

Grossman, S. J. and Stiglitz, J. E. (1980), ‘On the impossibility of informationally efficient markets’, *The American Economic Review* **70**(3), 393–408.

Hamid, S. A. and Iqbal, Z. (2004), ‘Using neural networks for forecasting volatility of s&p 500 index futures prices’, *Journal of Business Research* **57**(10), 1116–1125.

Hartzenburg, S. B. (2015), Global equity style anomalies, Master’s thesis, University of Cape Town.

Haykin, S. (1994), Neural networks, a comprehensive foundation, Technical report, Macmilan.

Hodnett, K. E. (2010), Analysis of the cross-section of equity returns on the JSE Securities Exchange based on linear and nonlinear modeling techniques, PhD thesis, University of Cape Town.

Jain, A. K., Mao, J. and Mohiuddin, K. (1996), ‘Artificial neural networks: A tutorial’, *Computer* **29**(3), 31–44.

Jensen, M. (1978), ‘Some anomalous evidence regarding market efficiency’, *Journal of Financial Economics* **6**(2/3), 95–101.

Kaastra, I. and Boyd, M. (1996), ‘Designing a neural network for forecasting financial and economic time series’, *Neurocomputing* **10**(3), 215–236.

Katz, J. O. (1992), 'Developing neural network forecasters for trading', *Technical Analysis of Stocks and Commodities* **10**(4), 160–168.

Klaussen, K. and Uhrig, J. W. (1994), Cash soybean price prediction with neural networks, *in* 'NCR-134 Conference on Applied Commodity Analysis, Price Forecasting and Market Risk Management Proceedings', pp. 56–65.

Kryzanowski, L., Galler, M. and Wright, D. W. (1993), 'Using artificial neural networks to pick stocks', *Financial Analysts Journal* **49**(4), 21–27.

Lawrence, S. and Giles, C. L. (2000), Overfitting and neural networks: conjugate gradient and backpropagation, *in* 'Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium', Vol. 1, IEEE, pp. 114–119.

Leshno, M. and Spector, Y. (1996), 'Neural network prediction analysis: The bankruptcy case', *Neurocomputing* **10**(2), 125–147.

Malkiel, B. G. (1999), *A random walk down Wall Street: including a life-cycle guide to personal investing*, WW Norton & Company.

Masters, T. (1993), *Practical neural network recipes in C++*, Morgan Kaufmann.

McCulloch, W. S. and Pitts, W. (1943), 'A logical calculus of the ideas immanent in nervous activity', *The bulletin of mathematical biophysics* **5**(4), 115–133.

Moya, J. G. (2011), Integration of the information in complex neural networks with noise, PhD thesis, Universitat Politècnica de Catalunya. Escola Tècnica Superior d'Enginyeries Industrial i Aeronàutica de Terrassa.

Nygren, K. (2004), 'Stock prediction–a neural network approach', *Royal Instiute of Technology* pp. 1–34.

Osborne, J. and Waters, E. (2002), 'Four assumptions of multiple regression that researchers should always test', *Practical assessment, research & evaluation* **8**(2), 1–9.

Shachmurove, Y. and Witkowska, D. (2000), 'Utilizing artificial neural network model to predict stock markets', *University of Pennsylvania, Center for Analytic Research in Economics and the Social Sciences* pp. 1–25.

Siganos, D. and Stergiou, C. (1996), ‘Neural networks, the human brain, and learning’, *Imperial College London: Surveys and Presentations in Information Systems Engineering* .
URL: *http://srii.sou.edu.ge/neural-networks.pdf*

Somasundaram, R. and Nedunchezhian, R. (2011), ‘Evaluation of three simple imputation methods for enhancing preprocessing of data with missing values’, *International Journal of Computer Applications* **21**(10), 14–19.

Svozil, D., Kvasnicka, V. and Pospichal, J. (1997), ‘Introduction to multi-layer feed-forward neural networks’, *Chemometrics and Intelligent Laboratory Systems* **39**(1), 43–62.

Tu, J. V. (1996), ‘Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes’, *Journal of clinical epidemiology* **49**(11), 1225–1231.

Uysal, M. and El Roubi, M. S. (1999), ‘Artificial neural networks versus multiple regression in tourism demand analysis’, *Journal of Travel Research* **38**(2), 111–118.

Werbos, P. (1974), Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences, PhD thesis, Harvard University.

Williams, M. N., Grajales, C. A. G. and Kurkiewicz, D. (2013), ‘Assumptions of multiple regression: Correcting two misconceptions’, *Practical Assessment, Research, and Evaluation* **18**(11).
URL: *https://scholarworks.umass.edu/pare/vol18/iss1/11*

Wong, B. K. and Selvi, Y. (1998), ‘Neural network applications in finance: A review and analysis of literature (1990–1996)’, *Information & Management* **34**(3), 129–139.

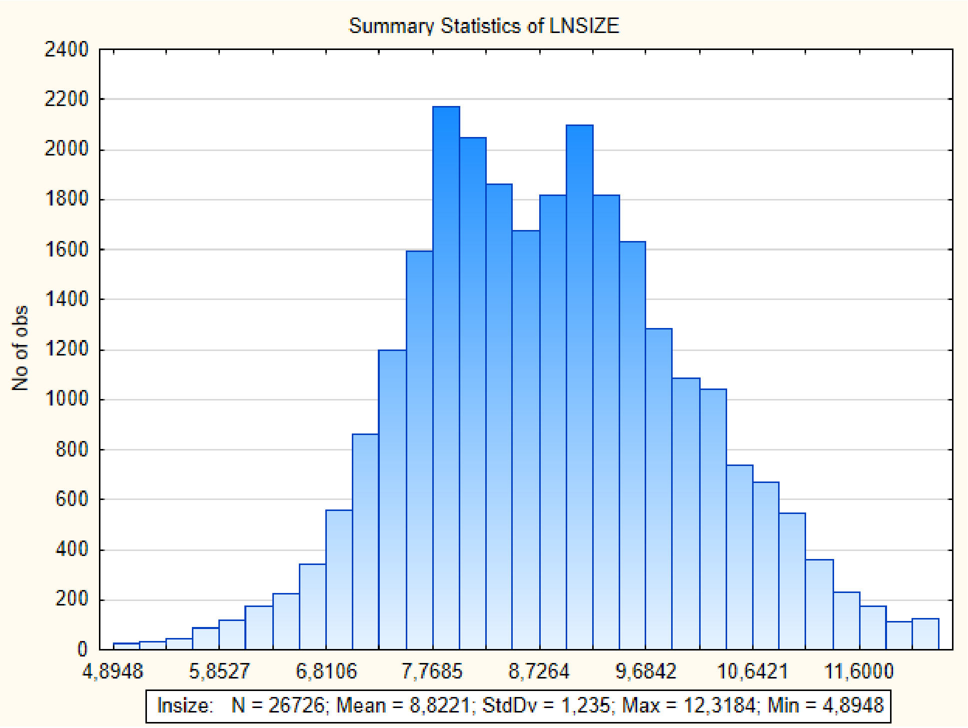
Yoon, Y. and Swales, G. (1991), Predicting stock price performance: A neural network approach, *in* ‘Proceedings of the twenty-fourth annual Hawaii international conference on system sciences’, Vol. 4, IEEE, pp. 156–162.

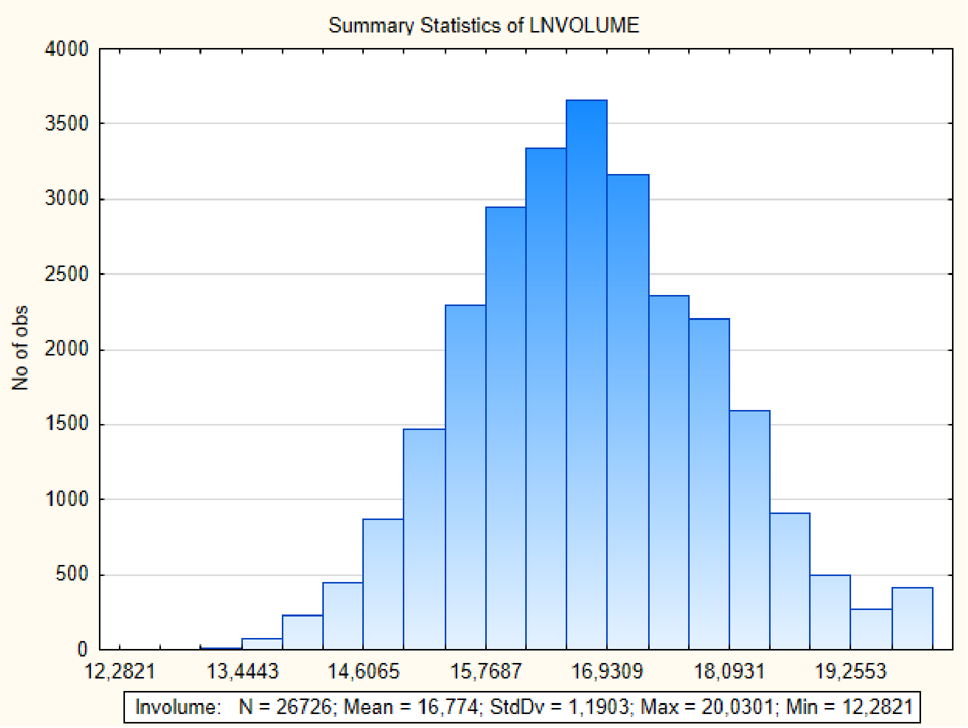
Zhang, G., Patuwo, B. E. and Hu, M. Y. (1998), ‘Forecasting with artificial neural networks:: The state of the art’, *International journal of forecasting* **14**(1), 35–62.

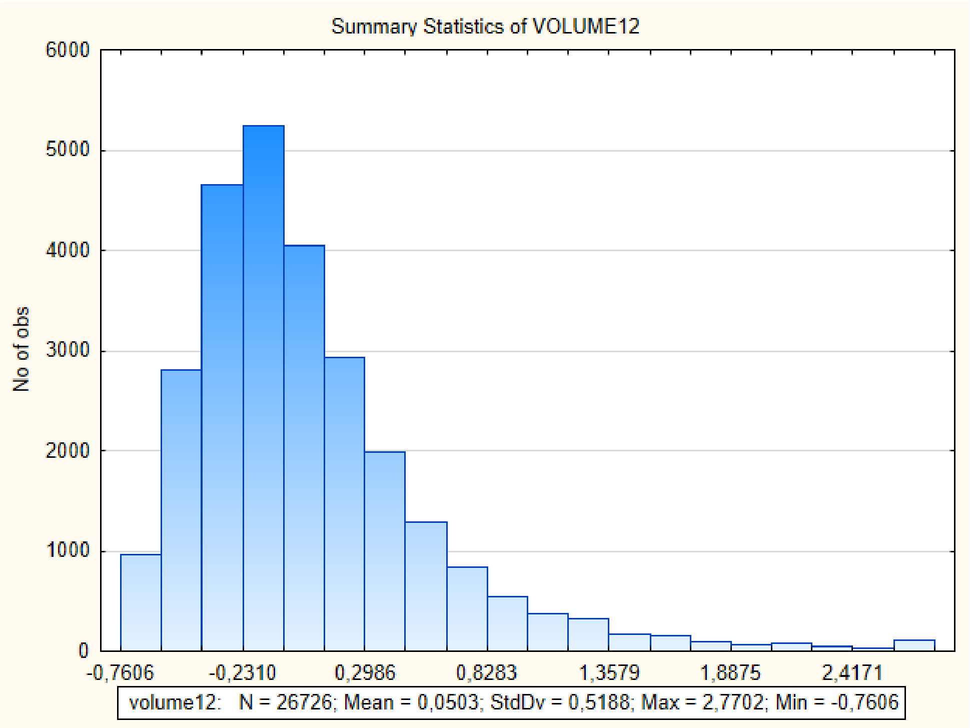
Appendices

Appendix A: Descriptive Statistics for Input Factors

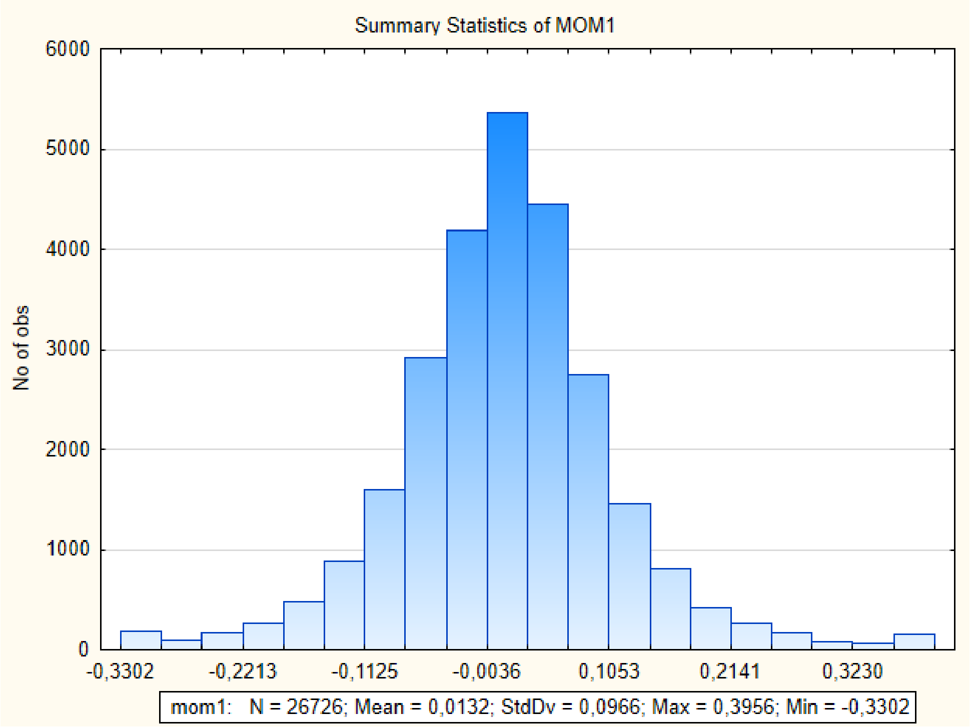
Size and Liquidity Factors

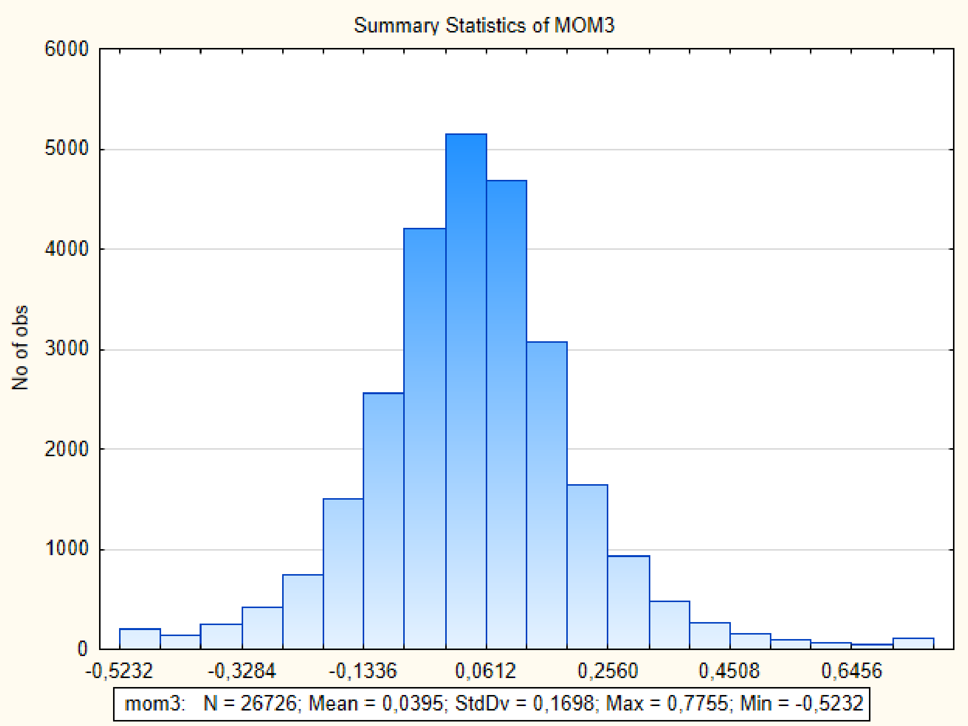


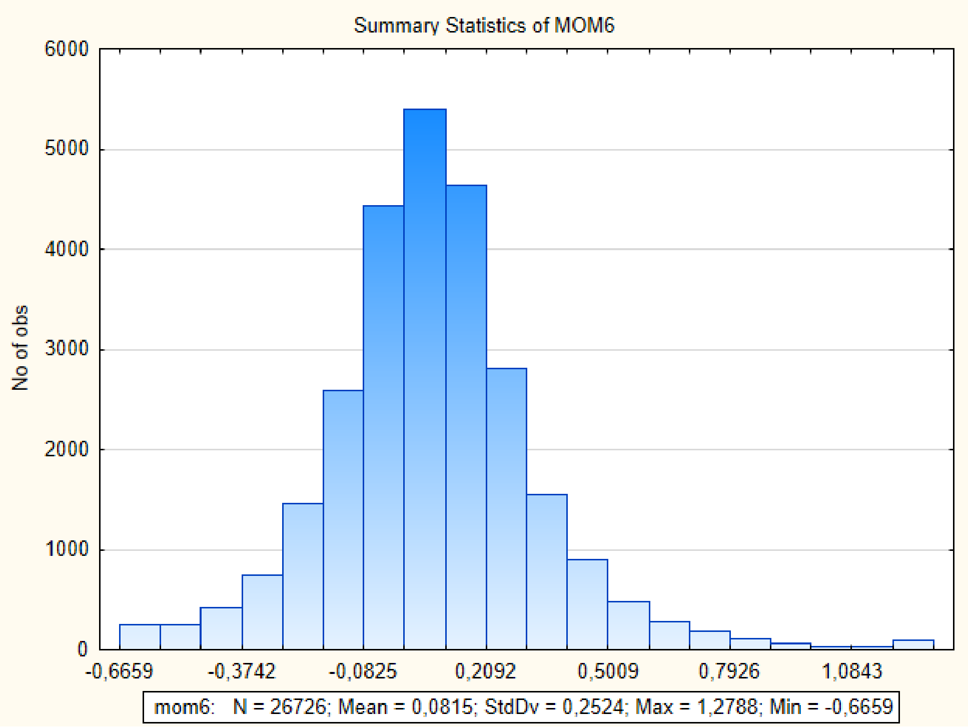




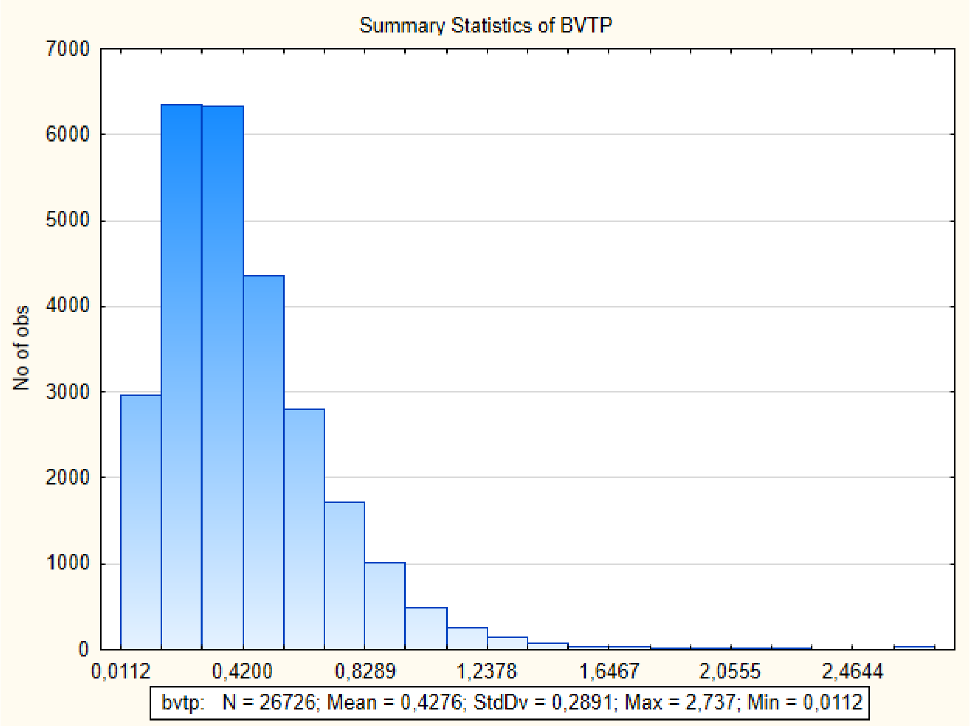
Momentum Factors

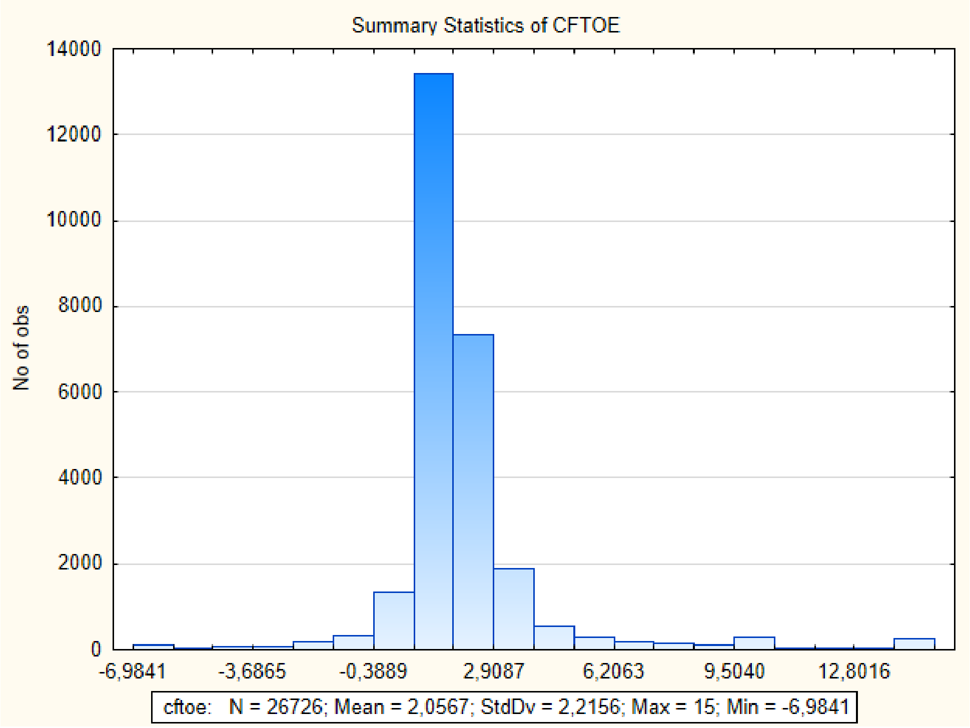


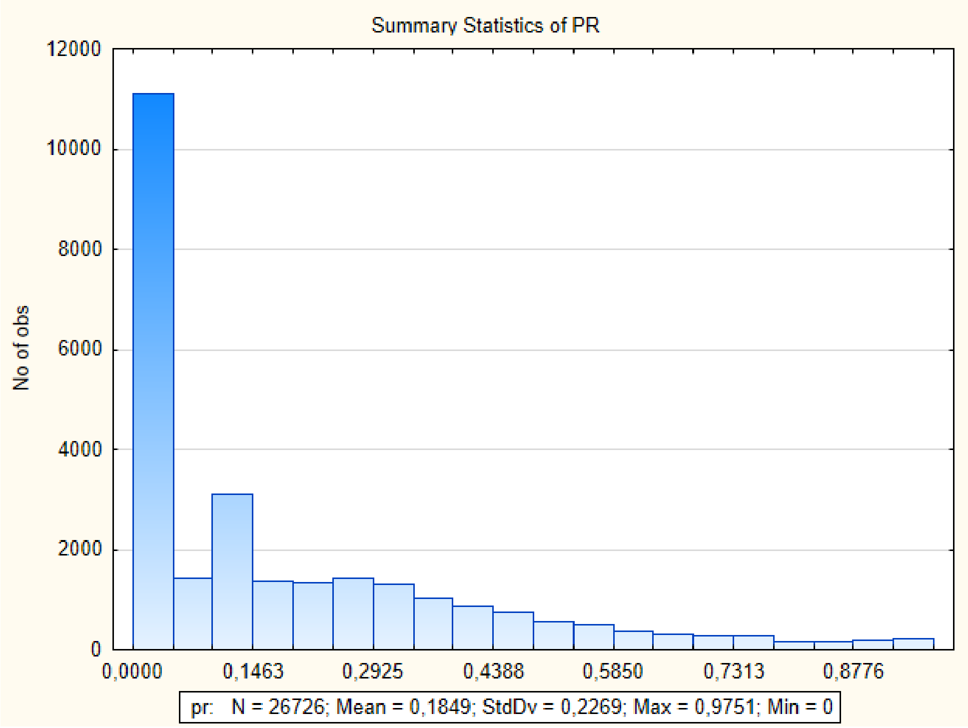

Summary Statistics of MOM3
No of obs
6000
5000
4000
3000
2000
1000
0
-0,5232
-0,3284
-0,1336
0,0612
0,2560
0,4508
0,6456
mom3: N = 26726; Mean = 0,0395; StdDv = 0,1698; Max = 0,7755; Min = -0,5232

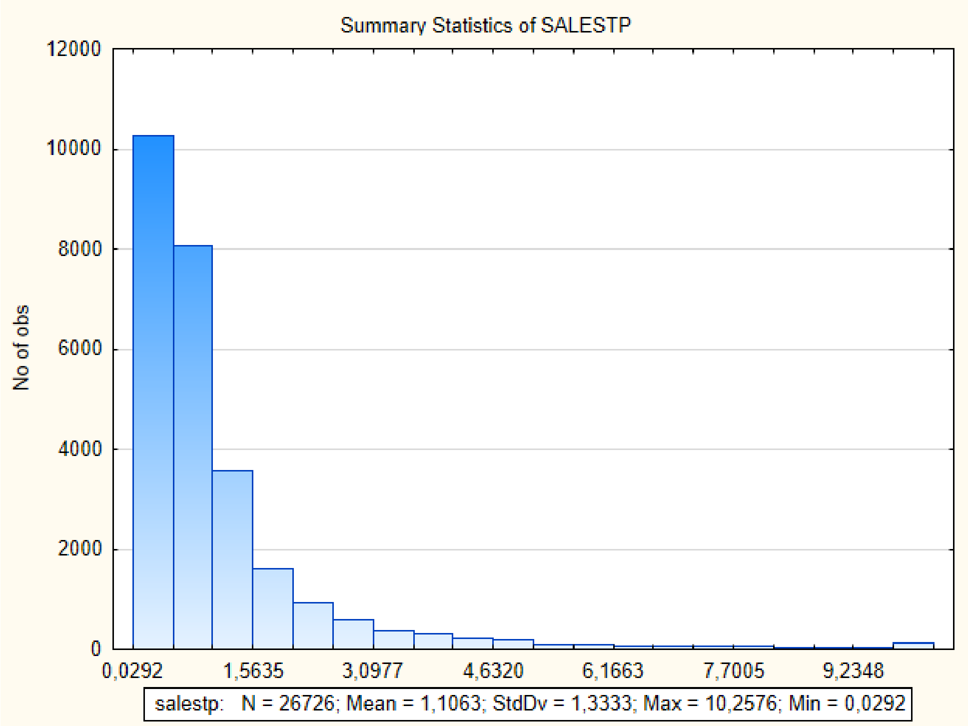

Summary Statistics of MOM6
No of obs
6000
5000
4000
3000
2000
1000
0
-0,6659
-0,3742
-0,0825
0,2092
0,5009
0,7926
1,0843
mom6: N = 26726; Mean = 0,0815; StdDv = 0,2524; Max = 1,2788; Min = -0,6659

Value Factors

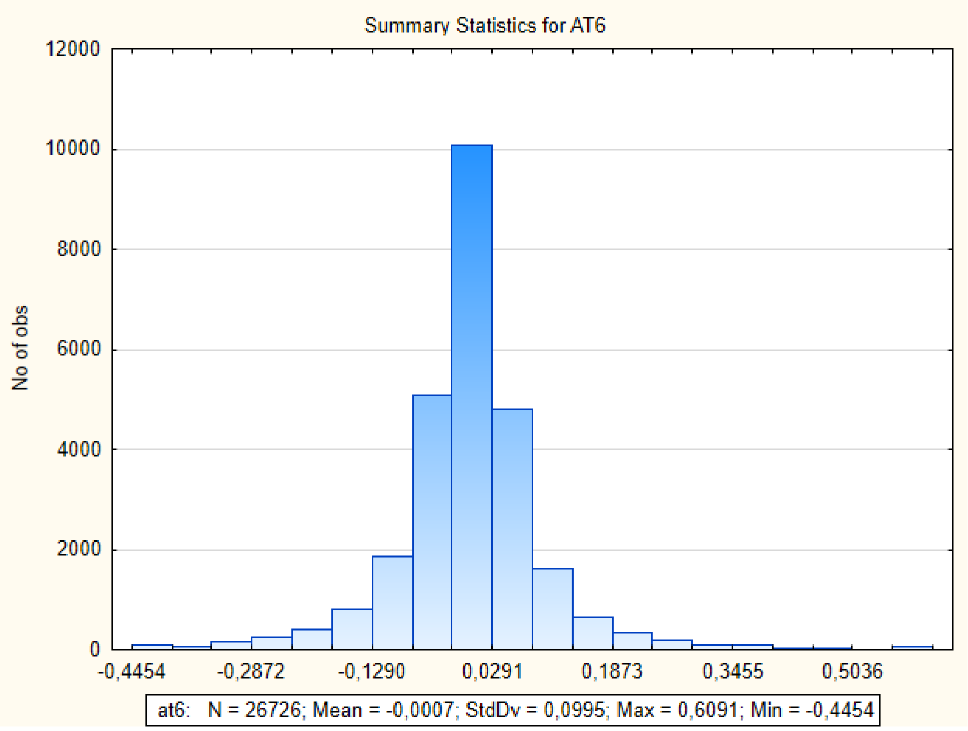


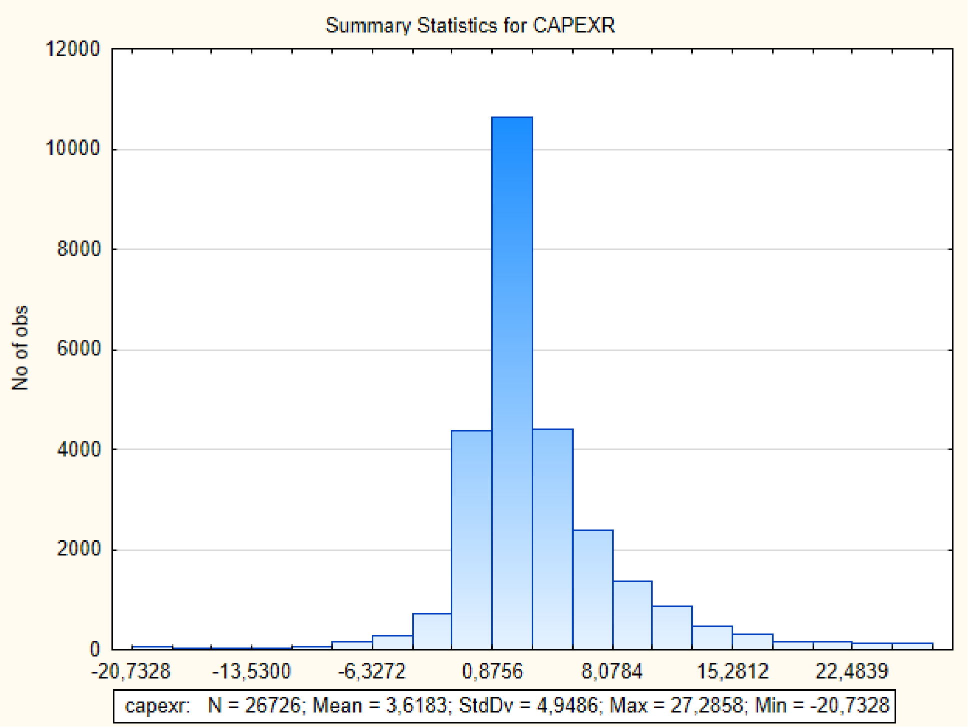


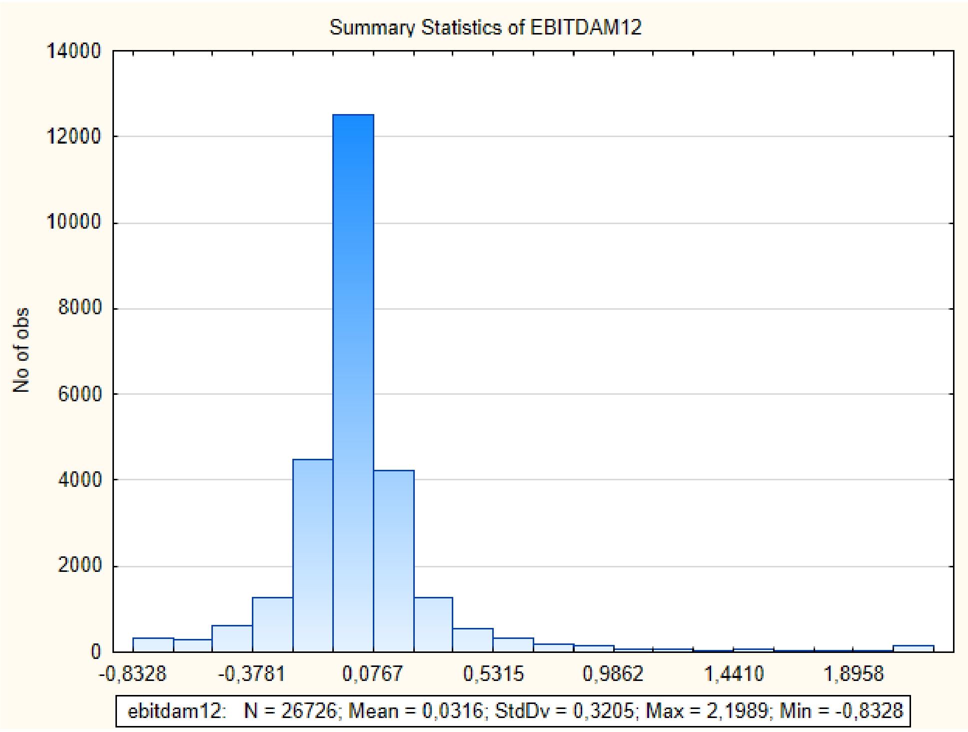

Summary Statistics of PR
No of obs
12000
10000
8000
6000
4000
2000
0
0,0000
0,1463
0,2925
0,4388
0,5850
0,7313
0,8776
pr: N = 26726; Mean = 0,1849; StdDv = 0,2269; Max = 0,9751; Min = 0

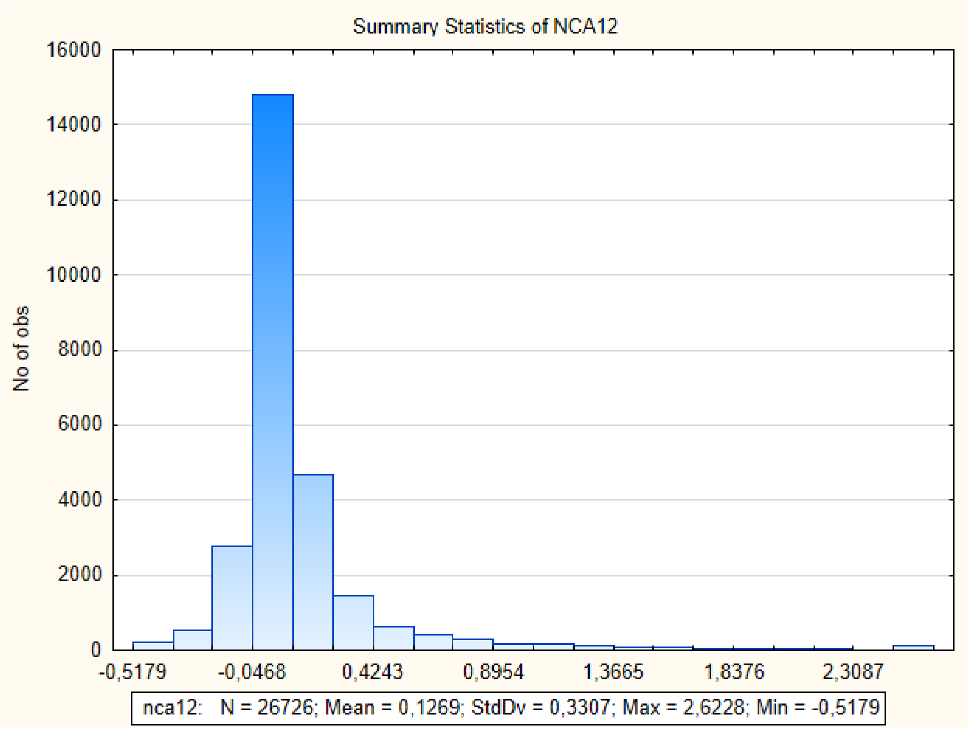

Summary Statistics of SALESTP
No of obs
12000
10000
8000
6000
4000
2000
0
0,0292
1,5635
3,0977
4,6320
6,1663
7,7005
9,2348
salestp: N = 26726; Mean = 1,1063; StdDv = 1,3333; Max = 10,2576; Min = 0,0292

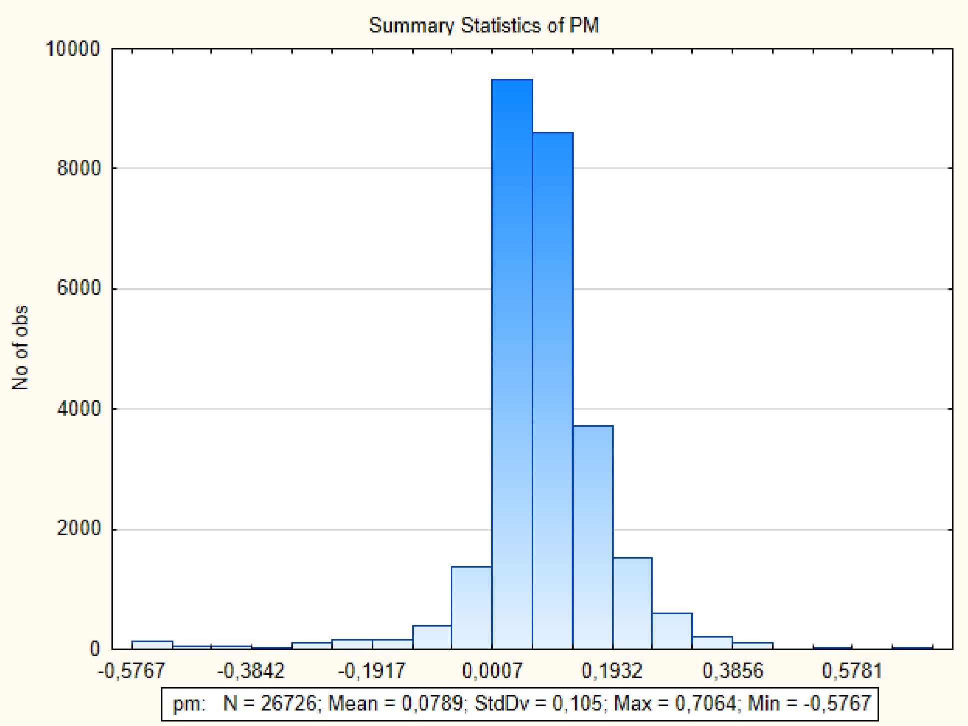
Growth Factors

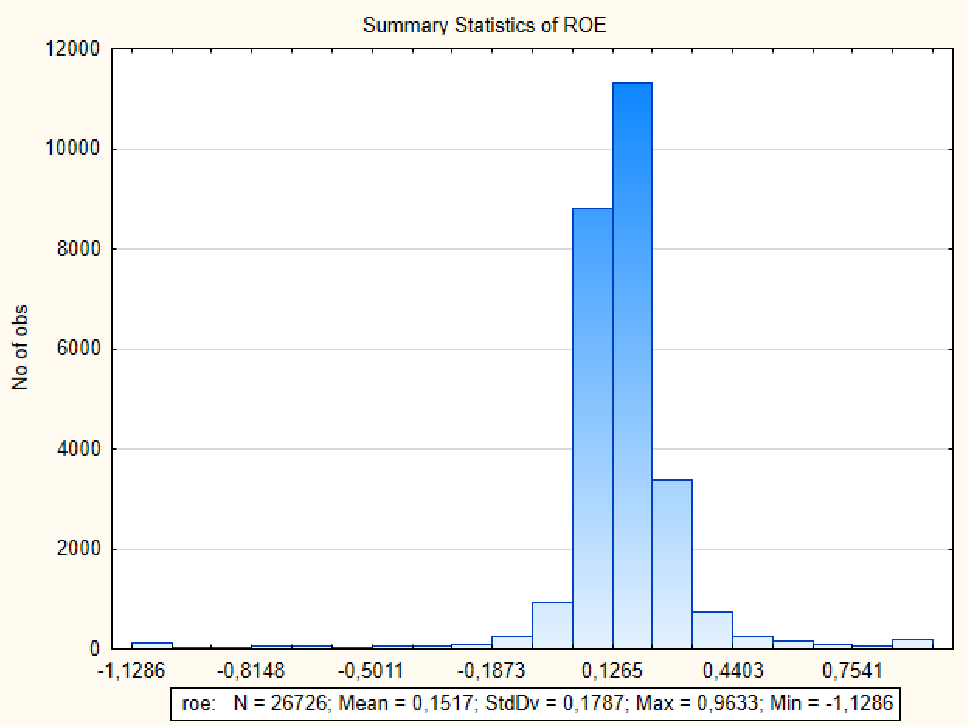


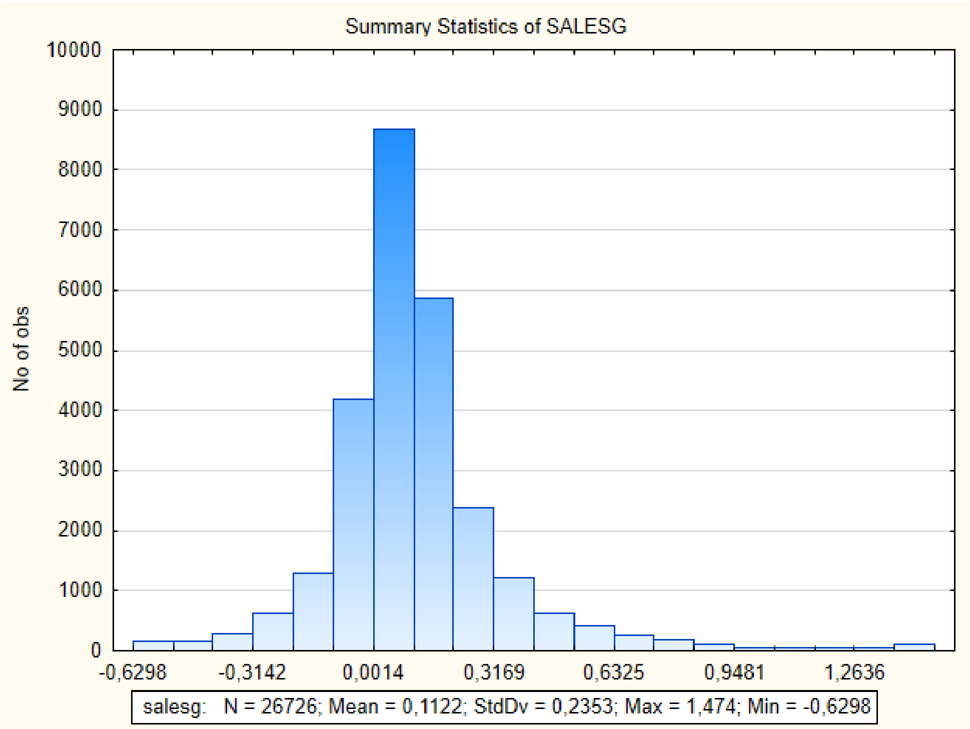



Summary Statistics of EBITDAM12
No of obs
14000
12000
10000
8000
6000
4000
2000
0
-0,8328
-0,3781
0,0767
0,5315
0,9862
1,4410
1,8958
ebitdam12: N = 26726; Mean = 0,0316; StdDv = 0,3205; Max = 2,1989; Min = -0,8328

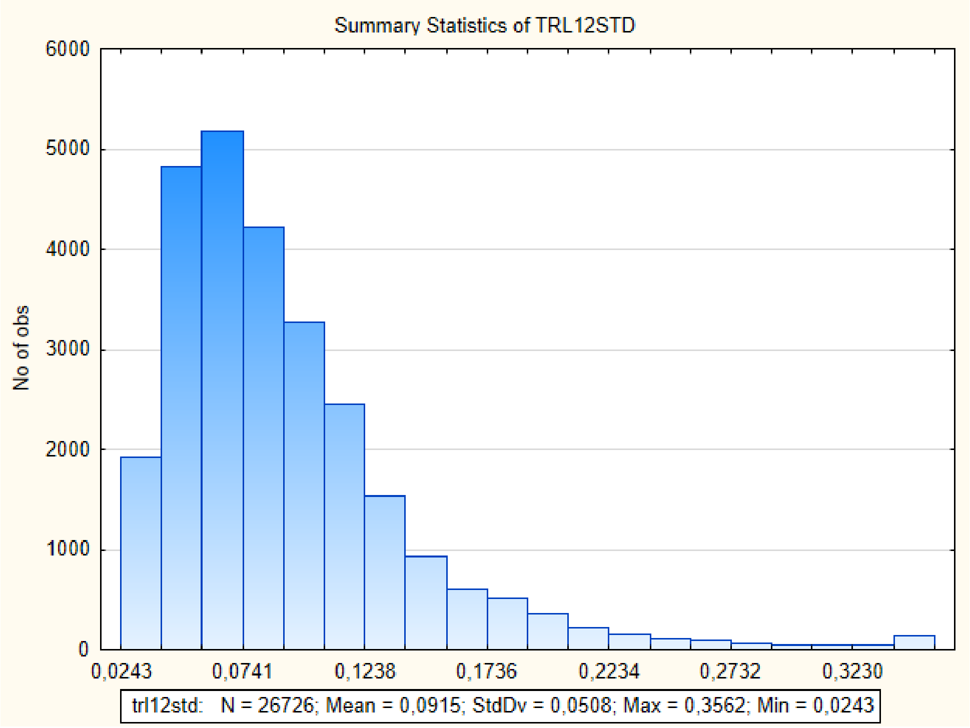

Summary Statistics of NCA12
No of obs
16000
14000
12000
10000
8000
6000
4000
2000
0
-0,5179
-0,0468
0,4243
0,8954
1,3665
1,8376
2,3087
nca12: N = 26726; Mean = 0,1269; StdDv = 0,3307; Max = 2,6228; Min = -0,5179


Summary Statistics of PM
No of obs
10000
8000
6000
4000
2000
0
-0,5767
-0,3842
-0,1917
0,0007
0,1932
0,3856
0,5781
pm: N = 26726; Mean = 0,0789; StdDv = 0,105; Max = 0,7064; Min = -0,5767

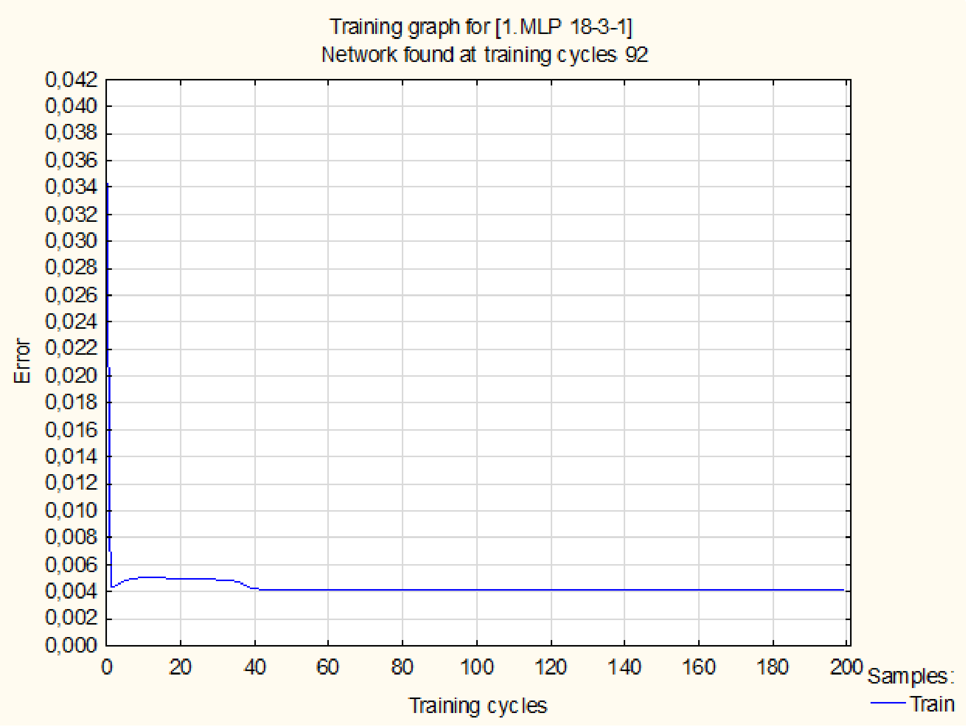

Summary Statistics of ROE
No of obs
12000
10000
8000
6000
4000
2000
0
-1,1286
-0,8148
-0,5011
-0,1873
0,1265
0,4403
0,7541
roe: N = 26726; Mean = 0,1517; StdDv = 0,1787; Max = 0,9633; Min = -1,1286

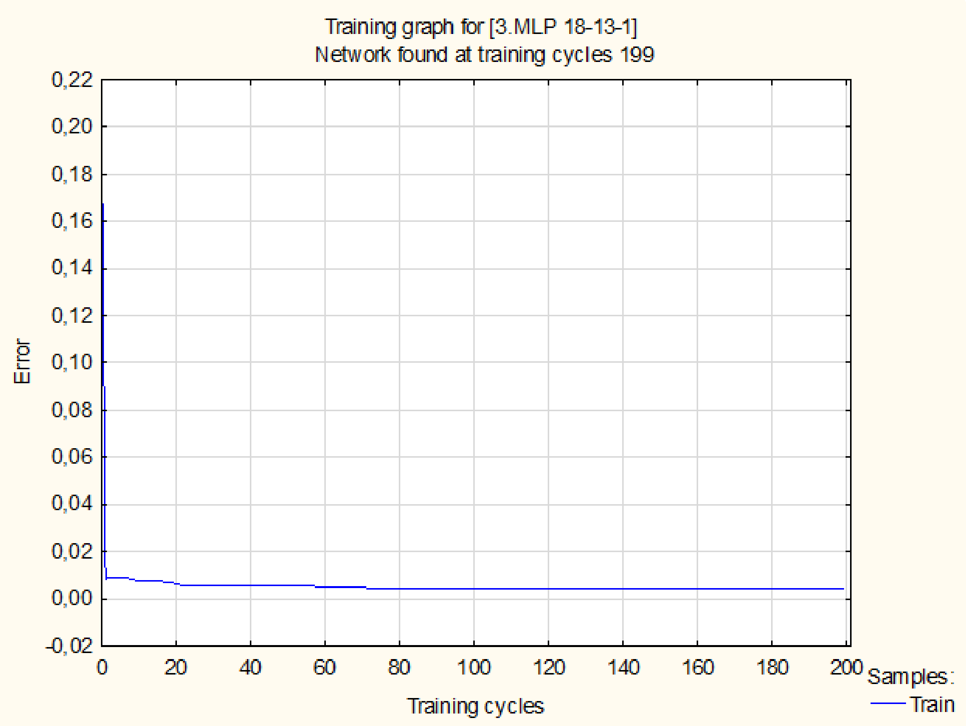


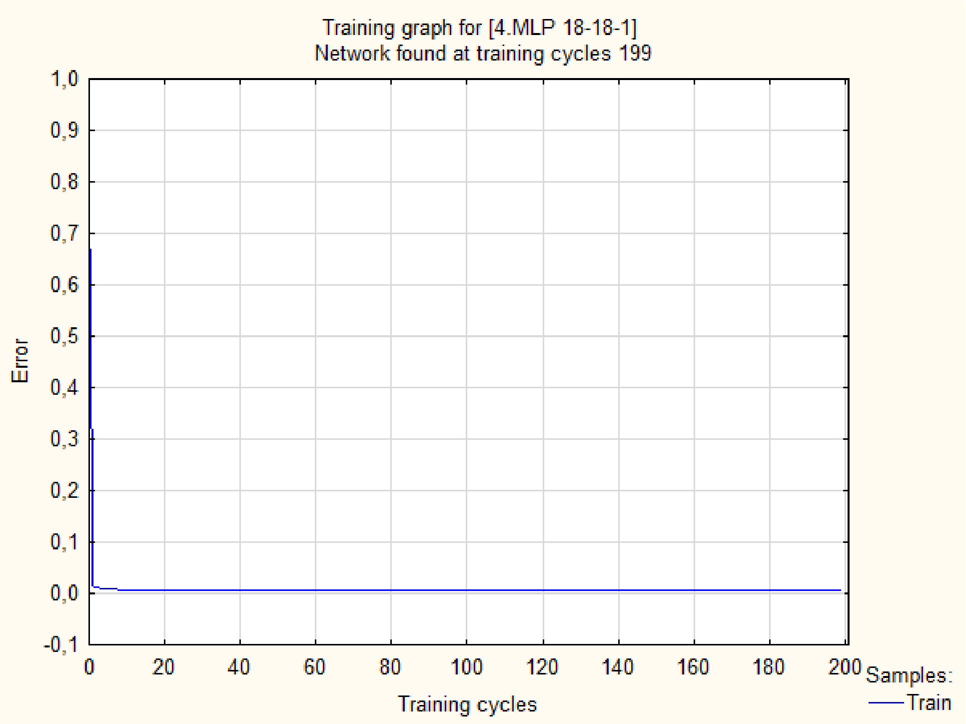
Risk Factors



Appendix B: Training Error Graphs




Training graph for [3.MLP 18-13-1]
Network found at training cycles 199
Error
0,22
0,20
0,18
0,16
0,14
0,12
0,10
0,08
0,06
0,04
0,02
0,00
-0,02
0
20
40
60
80
100
120
140
160
180
200
Training cycles
Samples:
Train


Training graph for [4.MLP 18-18-1]
Network found at training cycles 199
Error
1,0
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0,0
-0,1
0
20
40
60
80
100
120
140
160
180
200
Training cycles
Samples:
Train


Training graph for [5.MLP 18-30-1]
Network found at training cycles 199
Error
0,10
0,09
0,08
0,07
0,06
0,05
0,04
0,03
0,02
0,01
0,00
-0,01
0
20
40
60
80
100
120
140
160
180
200
Training cycles
Samples:
Train

Appendix C: Structure and Connection Weights for MLP 18-13-1

Connections	Weight Values
at6 --> hidden neuron 1	-0,11129
bvtp --> hidden neuron 1	0,46157
capexr --> hidden neuron 1	-0,56954
cftoe --> hidden neuron 1	0,11
ebitdam12 --> hidden neuron 1	0,1524
lnsize --> hidden neuron 1	-0,11013
lnvolume --> hidden neuron 1	-0,32457
mom1 --> hidden neuron 1	-0,2796
mom3 --> hidden neuron 1	0,32999
mom6 --> hidden neuron 1	-0,08094
nca12 --> hidden neuron 1	-0,05301
pm --> hidden neuron 1	-0,0863
pr --> hidden neuron 1	0,18512
roe --> hidden neuron 1	0,37784
salesg --> hidden neuron 1	0,25478
salestp --> hidden neuron 1	-0,20893
trl12std --> hidden neuron 1	0,24824
volume12 --> hidden neuron 1	0,12247

Connections	Weight Values
at6 --> hidden neuron 2	0,12449
bvtp --> hidden neuron 2	0,14079
capexr --> hidden neuron 2	0,30681
cftoe --> hidden neuron 2	-0,2562
ebitdam12 --> hidden neuron 2	-0,23244
lnsize --> hidden neuron 2	0,00469
lnvolume --> hidden neuron 2	-0,05998
mom1 --> hidden neuron 2	0,00323
mom3 --> hidden neuron 2	0,15941
mom6 --> hidden neuron 2	-0,31965
nca12 --> hidden neuron 2	0,13454
pm --> hidden neuron 2	-0,11088
pr --> hidden neuron 2	-0,05277
roe --> hidden neuron 2	0,16622
salesg --> hidden neuron 2	0,24285
salestp --> hidden neuron 2	0,18075
trl12std --> hidden neuron 2	0,14853
volume12 --> hidden neuron 2	0,0159

Connections	Weight Values
at6 --> hidden neuron 3	0,18131
bvtp --> hidden neuron 3	-0,34961
capexr --> hidden neuron 3	-0,30886
cftoe --> hidden neuron 3	-0,07484
ebitdam12 --> hidden neuron 3	-0,44378
lnsize --> hidden neuron 3	0,19227
lnvolume --> hidden neuron 3	-0,09813
mom1 --> hidden neuron 3	0,06153
mom3 --> hidden neuron 3	-0,42566
mom6 --> hidden neuron 3	0,55788
nca12 --> hidden neuron 3	-0,53217
pm --> hidden neuron 3	-0,12355
pr --> hidden neuron 3	0,20065
roe --> hidden neuron 3	-0,15442
salesg --> hidden neuron 3	0,3002
salestp --> hidden neuron 3	-0,05176
trl12std --> hidden neuron 3	0,15901
volume12 --> hidden neuron 3	-0,00038

Connections	Weight Values
at6 --> hidden neuron 4	0,44873
bvtp --> hidden neuron 4	-0,00162
capexr --> hidden neuron 4	-0,08844
cftoe --> hidden neuron 4	0,22971
ebitdam12 --> hidden neuron 4	-0,09331
lnsize --> hidden neuron 4	-0,1269
lnvolume --> hidden neuron 4	0,30475
mom1 --> hidden neuron 4	-0,06528
mom3 --> hidden neuron 4	0,1796
mom6 --> hidden neuron 4	-0,19494
nca12 --> hidden neuron 4	0,11036
pm --> hidden neuron 4	-0,28967
pr --> hidden neuron 4	0,14931
roe --> hidden neuron 4	0,31423
salesg --> hidden neuron 4	0,03155
salestp --> hidden neuron 4	0,10362
trl12std --> hidden neuron 4	0,25272
volume12 --> hidden neuron 4	-0,23175

Connections	Weight Values
at6 --> hidden neuron 5	-0,08989
bvtp --> hidden neuron 5	0,47945
capexr --> hidden neuron 5	0,72188
cftoe --> hidden neuron 5	-0,04034
ebitdam12 --> hidden neuron 5	0,24672
lnsize --> hidden neuron 5	-0,45481
lnvolume --> hidden neuron 5	0,53118
mom1 --> hidden neuron 5	0,05429
mom3 --> hidden neuron 5	-0,308
mom6 --> hidden neuron 5	0,10652
nca12 --> hidden neuron 5	-0,00188
pm --> hidden neuron 5	-0,11026
pr --> hidden neuron 5	0,03403
roe --> hidden neuron 5	-0,23767
salesg --> hidden neuron 5	-0,2203
salestp --> hidden neuron 5	-0,32351
trl12std --> hidden neuron 5	-0,13327
volume12 --> hidden neuron 5	0,05047

Connections	Weight Values
at6 --> hidden neuron 6	-0,02048
bvtp --> hidden neuron 6	0,14476
capexr --> hidden neuron 6	0,30713
cftoe --> hidden neuron 6	0,02271
ebitdam12 --> hidden neuron 6	-0,10168
lnsize --> hidden neuron 6	-0,83046
lnvolume --> hidden neuron 6	0,58119
mom1 --> hidden neuron 6	-0,03473
mom3 --> hidden neuron 6	-0,05527
mom6 --> hidden neuron 6	-0,11335
nca12 --> hidden neuron 6	-0,1363
pm --> hidden neuron 6	-0,54176
pr --> hidden neuron 6	0,12657
roe --> hidden neuron 6	0,63465
salesg --> hidden neuron 6	-0,02565
salestp --> hidden neuron 6	0,02237
trl12std --> hidden neuron 6	-0,06303
volume12 --> hidden neuron 6	-0,11438

Connections	Weight Values
at6 --> hidden neuron 7	0,35131
bvtp --> hidden neuron 7	-1,33777
capexr --> hidden neuron 7	0,30457
cftoe --> hidden neuron 7	-0,34385
ebitdam12 --> hidden neuron 7	0,03584
lnsize --> hidden neuron 7	0,94249
lnvolume --> hidden neuron 7	-0,93329
mom1 --> hidden neuron 7	-0,08469
mom3 --> hidden neuron 7	-0,18247
mom6 --> hidden neuron 7	-0,16194
nca12 --> hidden neuron 7	0,17116
pm --> hidden neuron 7	-0,43214
pr --> hidden neuron 7	-0,13274
roe --> hidden neuron 7	0,3722
salesg --> hidden neuron 7	-0,17277
salestp --> hidden neuron 7	-0,04631
trl12std --> hidden neuron 7	0,34474
volume12 --> hidden neuron 7	-0,11116

Connections	Weight Values
at6 --> hidden neuron 8	-0,1023
bvtp --> hidden neuron 8	-1,18679
capexr --> hidden neuron 8	-0,07274
cftoe --> hidden neuron 8	0,56342
ebitdam12 --> hidden neuron 8	0,13831
lnsize --> hidden neuron 8	0,63197
lnvolume --> hidden neuron 8	-0,34739
mom1 --> hidden neuron 8	0,06482
mom3 --> hidden neuron 8	0,0574
mom6 --> hidden neuron 8	-0,21432
nca12 --> hidden neuron 8	0,05537
pm --> hidden neuron 8	-0,0562
pr --> hidden neuron 8	-0,24271
roe --> hidden neuron 8	0,02457
salesg --> hidden neuron 8	-0,03393
salestp --> hidden neuron 8	-0,4572
trl12std --> hidden neuron 8	0,03643
volume12 --> hidden neuron 8	0,09927

Connections	Weight Values
at6 --> hidden neuron 9	-0,28882
bvtp --> hidden neuron 9	-0,47347
capexr --> hidden neuron 9	0,70476
cftoe --> hidden neuron 9	-0,17281
ebitdam12 --> hidden neuron 9	0,24694
lnsize --> hidden neuron 9	-0,62282
lnvolume --> hidden neuron 9	0,58653
mom1 --> hidden neuron 9	-0,04326
mom3 --> hidden neuron 9	-0,05008
mom6 --> hidden neuron 9	-0,16147
nca12 --> hidden neuron 9	-0,50918
pm --> hidden neuron 9	0,11497
pr --> hidden neuron 9	-0,30333
roe --> hidden neuron 9	-0,2032
salesg --> hidden neuron 9	-0,27788
salestp --> hidden neuron 9	0,6493
trl12std --> hidden neuron 9	-0,15286
volume12 --> hidden neuron 9	-0,25517

Connections	Weight Values
at6 --> hidden neuron 10	0,16882
bvtp --> hidden neuron 10	0,14553
capexr --> hidden neuron 10	0,24418
cftoe --> hidden neuron 10	0,44728
ebitdam12 --> hidden neuron 10	-0,32025
lnsize --> hidden neuron 10	-0,11515
lnvolume --> hidden neuron 10	-0,0418
mom1 --> hidden neuron 10	0,37138
mom3 --> hidden neuron 10	-0,33546
mom6 --> hidden neuron 10	-0,02802
nca12 --> hidden neuron 10	-0,6801
pm --> hidden neuron 10	0,43112
pr --> hidden neuron 10	-0,04732
roe --> hidden neuron 10	-0,20852
salesg --> hidden neuron 10	0,0385
salestp --> hidden neuron 10	0,07334
trl12std --> hidden neuron 10	0,26004
volume12 --> hidden neuron 10	-0,15021

Connections	Weight Values
at6 --> hidden neuron 11	0,09722
bvtp --> hidden neuron 11	0,57848
capexr --> hidden neuron 11	0,28405
cftoe --> hidden neuron 11	-0,404
ebitdam12 --> hidden neuron 11	-0,35751
lnsize --> hidden neuron 11	0,40248
lnvolume --> hidden neuron 11	-0,16351
mom1 --> hidden neuron 11	-0,0462
mom3 --> hidden neuron 11	-0,15024
mom6 --> hidden neuron 11	-0,02609
nca12 --> hidden neuron 11	-0,35088
pm --> hidden neuron 11	0,18768
pr --> hidden neuron 11	-0,26375
roe --> hidden neuron 11	0,19814
salesg --> hidden neuron 11	-0,04443
salestp --> hidden neuron 11	-0,22917
trl12std --> hidden neuron 11	0,15745
volume12 --> hidden neuron 11	0,00669

Connections	Weight Values
at6 --> hidden neuron 12	-0,09797
bvtp --> hidden neuron 12	0,73356
capexr --> hidden neuron 12	-0,20317
cftoe --> hidden neuron 12	0,20431
ebitdam12 --> hidden neuron 12	0,20192
lnsize --> hidden neuron 12	-0,3127
lnvolume --> hidden neuron 12	0,38154
mom1 --> hidden neuron 12	0,10492
mom3 --> hidden neuron 12	-0,06973
mom6 --> hidden neuron 12	0,31298
nca12 --> hidden neuron 12	0,07982
pm --> hidden neuron 12	-0,20651
pr --> hidden neuron 12	-0,02688
roe --> hidden neuron 12	0,21407
salesg --> hidden neuron 12	0,15134
salestp --> hidden neuron 12	-0,05417
trl12std --> hidden neuron 12	0,01075
volume12 --> hidden neuron 12	-0,00414

Connections	Weight Values
at6 --> hidden neuron 13	-0,09407
bvtp --> hidden neuron 13	-0,15718
capexr --> hidden neuron 13	-0,37981
cftoe --> hidden neuron 13	-0,54151
ebitdam12 --> hidden neuron 13	-0,46204
lnsize --> hidden neuron 13	0,65644
lnvolume --> hidden neuron 13	0,11286
mom1 --> hidden neuron 13	-0,2087
mom3 --> hidden neuron 13	0,08608
mom6 --> hidden neuron 13	0,34448
nca12 --> hidden neuron 13	-0,10542
pm --> hidden neuron 13	0,50526
pr --> hidden neuron 13	0,30157
roe --> hidden neuron 13	-0,1102
salesg --> hidden neuron 13	0,28894
salestp --> hidden neuron 13	0,16259
trl12std --> hidden neuron 13	0,05823
volume12 --> hidden neuron 13	-0,04812

Connections	Weight Values
input bias --> hidden neuron 1	0,08282
input bias --> hidden neuron 2	-0,18242
input bias --> hidden neuron 3	0,30143
input bias --> hidden neuron 4	-0,4469
input bias --> hidden neuron 5	-0,16201
input bias --> hidden neuron 6	-0,08194
input bias --> hidden neuron 7	0,22537
input bias --> hidden neuron 8	-0,09107
input bias --> hidden neuron 9	0,20252
input bias --> hidden neuron 10	-0,21884
input bias --> hidden neuron 11	-0,11509
input bias --> hidden neuron 12	-0,38444
input bias --> hidden neuron 13	-0,17967

Connections	Weight Values
hidden neuron 1 --> returnsfwd	0,02756
hidden neuron 2 --> returnsfwd	0,07831
hidden neuron 3 --> returnsfwd	-0,07949
hidden neuron 4 --> returnsfwd	0,05545
hidden neuron 5 --> returnsfwd	0,00766
hidden neuron 6 --> returnsfwd	0,08699
hidden neuron 7 --> returnsfwd	-0,02056
hidden neuron 8 --> returnsfwd	-0,09055
hidden neuron 9 --> returnsfwd	0,08831
hidden neuron 10 --> returnsfwd	0,02396
hidden neuron 11 --> returnsfwd	0,03505
hidden neuron 12 --> returnsfwd	0,01444
hidden neuron 13 --> returnsfwd	-0,09608
hidden bias --> returnsfwd	0,69673

Appendix D: Predicted vs Actual Returns Regression Output for MLP 18-13-1

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0,054030613
R Square	0,002919307
Adjusted R Square	0,002783797
Standard Error	0,075887825
Observations	7360

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	1	0,124066201	0,124066	21,54315	3,51949E-06
Residual	7358	42,37444216	0,005759		
Total	7359	42,49850836			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*
Intercept	0,014582742	0,00096106	15,1736	3,14E-51	0,012698788	0,016466695
Predicted Returns	0,101597183	0,021889056	4,64146	3,52E-06	0,058688364	0,144506003