The University Of Sheffield.

CITY College An International Faculty Of The University.

The University of Sheffield International Faculty

Department of Business Administration and Economics

Using artificial neural networks to test the efficient market Hypothesis- The Greek stock market case

Spyridon Vergos

January 2018

The University Of Sheffield.

CITY College An International Faculty Of The University.

The University of Sheffield

International Faculty

Department of Business Administration and Economics

Using artificial neural networks to test the efficient market Hypothesis- The Greek stock market case

This report is submitted in partial fulfillment for the degree of Master of Science in Banking and Finance by

Spyridon Vergos

January 2018

Supervisor

Dr. Athanasios Fassas

The University Of Sheffield.
CITY College An International Faculty Of The University.

Using artificial neural networks to test the efficient market Hypothesis- The Greek stock market case

Spyridon Vergos

Abstract

The fields of big data analysis and deep learning are developing fast recently mainly due to the vast availability of data sets online and the unpreceded cheap computational power of cloud services. LSTM Neural Networks is a specialized function approximator for time-series problems. It involves memory neurons that can overcome the training problems faced by classical Recurrent Neural Networks with very deep architectures. In this Thesis, we use an LSTM Neural Network to test the Efficient Market Hypothesis introduced by Eugene Fama by attempting to forecast the movement of the Greek Index Future FTSE/ATHEX Large Cap. Additionally, we compare the performance of our LSTM model performance with a Multilayered Perceptron with a time window, a traditional ARMA model using as variables autoregressive terms of our dataset and one common trading tactic namely the Moving Average CrossOver. The results showed that both Artificial Intelligence methods outperform the classic econometric (ARMA) model and the conventional trading tactic on prediction accuracy and trading performance. We conclude that the performed accuracy is impressive nevertheless it might be attributed to the good fitting of the model in the out of sample data and lack of exposure to LSTM of the particular market. Finally, we suggest ideas for further research.

Keywords: Algorithmic Trading, Efficient Market Hypothesis, Artificial Neural Networks.

The University Of Sheffield.

CITY College An International Faculty Of The University.

DECLARATION

All sentences or passages quoted in this dissertation from other people's work have been specifically acknowledgment by clear cross-referencing to author, work and ages(s).I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this dissertation and the degree examination as a whole.

Name .....

Signed …….. Date .....

Acknowledgements

I would like to express my sincere gratitude to everyone that helped and supported me the past two years. Especially my two families the one in Greece and the one in the U.S.A that where by my side during the darkest hours. Special thanks to Dr. Fassas who believed in my idea and accepted to supervise it.

Using artificial neural networks to test the efficient market Hypothesis- The Greek stock market case

2018

Contents

Glossary (Stegemann and Buenfeld, 1999)

Activation function … A bounded function of finite domain applied to the weighted and summed inputs to limit the amplitude of the output signal. For multi-layer networks, this must be a continuously differentiable function.

Artificial neural network … A class of flexible nonlinear regression and discriminant models, data reduction models and nonlinear dynamical systems consisting of a number of neurons (i.e., cells) interconnected in complex ways and often organized into layers.

Backpropagation … A method for computing the error dynamic feedback, gradient, i.e. the derivatives of the learning logic error function with respect to the weights, for a feedforward network.

Batch … The training data is divided into batches. Theoretically one can feed all the training data into a network at once, but practically it is limited to the computer's memory.

Bias … A weight parameter for an extra input whose activation is permanently set to +1.

Classification … assignment of inputs to discrete output classes.

Deep network … Artificial neural network with multiple hidden layers, i.e. more adjustable parameters/higher degrees of freedom.

Deep learning … Machine learning with deep neural networks.

Epoch … Each repeated entry of the full set of training patterns.

Error function … An expression which describes the difference between the computed and target output. Typically, the squared error, but sometimes linear error, absolute error or entropy.

Feedforward … Uni-directional transfer of information.

Layers … An arrangement of cells which process information in parallel, i.e., synchronously. Typically, one or more hidden layers are found between the input layer and the output layer.

Learning rate … A small parameter which sets the step size for adjustment of weights during training.

LSTM network…A recurrent neural network structure which identities autoregressive properties in the data of arbitrary length.

Machine learning…Methods in which an algorithm learns, i.e. optimizes parameters, in order to perform a task.

Network training…Optimizing the adjustable parameters of a neural network.

Neuron … a simple linear or nonlinear computing element that accepts one or more inputs computes a function thereof and may direct the result to one or more other cells.

Non-convex optimization …Optimizing functions which are nonconvex. When numerically optimizing non-convex functions, there is no guarantee that one arrives at the global minimum/maximum. Thus, programming algorithms for these types of problems require some thought.

Recurrent neural network…This network structure is different from a feedforward structure. Data is reprocessed by the hidden layers within the network.

Regularization …A term used in machine learning to describe the process of preventing overfitting of the training data.

Weights (w) & biases (b)…These are the adjustable parameters of a neural network. One can think of them as k and m in $y = kx + m$.

Perceptron… a class of feedforward artificial neural network.

Multilayered Perceptron…a fully connected feedforward error backpropagation neural network with at least one hidden layer.

List of Abbreviations

ADAM……Adaptive Moment Estimator.

AI ……..…….Artificial Intelligence.

ANN or NN …….. Artificial Neural Network.

API ……..… Application Programming Interface.

APT……..Arbitrage Pricing Theory.

ARMA……..Autoregressive Moving Average.

AWS……Amazon Web Services.

CAPM……..……Capital Asset Pricing Model.

EC2……..Amazon Elastic Computing Cloud.

EMH …… ……. Efficient Market Hypothesis.

FTSE/ATHEX…Financial Times-Stock Exchange- Athens Stock Exchange

HFT……..……..High-Frequency Trading.

LSTM …… …… Long Short-Term Memory.

MAC……Moving Average Crossover.

MAE……..…..Mean Average Error.

MLP…… Multilayered Perceptron.

MSE…….. Mean Squared Error.

OHLC……..Open, High, Low, Close.

RNN …….. Recurrent Neural Network.

List of Tables

List of Figures and Graphs

List of Formulas

Chapter1. Introduction.

Since antiquity, active trading strategies have been developed in order to exploit the market using momentum tactics and fundamental analysis (Shaede, 1989). This Thesis suggests a reconsideration of the famous Eugene Fama Efficient Market hypothesis (Fama,1970) taking into account the latest technological progress in the field of time series analysis.

Today, financial markets are a very crucial part of the economic and social structure of modern society. Financial activity influences the development and relationships of countries worldwide (Lin, Chiu and Lin, 2012). In Finance, the success of an investor depends on the quality of data he can have access to and the speed of his decision making and execution. This led financial analysis to be thoroughly studied through the fields of finance, engineering and mathematics in the previous decades (Yoo, Kim and Jan, 2005).

Financial time series prediction is considered today one of the most important challenges in the time series and machine learning fields (Tay and Cao, 2001). In the previous year's many approaches have been presented to predict financial time series and help in decision-making problems (Teixeira and Oliveira, 2010). The main approaches on forecasting financial time series are: statistical models and machine learning (Wang et al 2011). Traditional statistical methods assume in general that time series are generated from a linear process (Kumar and Murugan, 2013).

However empirical results showed that financial time series is much more complex, very noisy, dynamic, nonlinear, nonparametric and chaotic by default (Si and Yin, 2013). Machine learning methods are very efficient in capturing nonlinearities between variables in a

dataset even if the variables are not previously labeled (Atsalakis and Valavanis, 2009). In this thesis, we will attempt a research on forecasting of financial returns series by comparing an LSTM Recurrent Artificial Neural Network model to traditional investment forecasting strategies. Financial forecasting is considered a very controversial and demanding field. Thus, testing the weak form of Efficient Market hypothesis using a novel methodology like LSTM is very challenging.

1.1. Research Problem/query, research objectives

Since Eugene Fama wrote his Thesis and subsequent famous paper about Efficient Market Hypothesis (Fama,1970) there are significant technological advancements that allow interpreting "chaotic" datasets with unprecedented precision. The recent advancements in methodologies such as deep learning and Artificial Intelligence justify further research about the limitation of EMH theory. As a Master Thesis, we will challenge the weak form of efficient market hypothesis building a Long-Short Term deep learning neural network.

Research Queries:

- Has the Long-Short Term Memory Architecture Deep Neural Network written in python and using the Tensor-flow library the forecasting capability to challenge the weak form of Efficient Market Hypothesis using as data the daily returns of the Athens Stock Exchange Large Cap index for the period of 22 January 2001 to 20 October 2017?

- How does the Long-Short Term Memory Deep Neural Network compare to a Multilayered Perceptron using a time step window for the same dataset?

- How does a conventional time series ARMA model forecasting performance compare to the suggested neural network models?

- How does a trading strategy following the suggested neural network forecasts perform compared to following a moving average crossover forecast?

1.2. Thesis positioning

Despite Long-Sort Term Memory Recurrent Neural Networks have been in researcher's disposal for a long time now, their usage for testing the efficiency of stock markets globally and especially in the South-Eastern Mediterranean peninsula is not abundant in literature. Therefore, the accessibility to data and the computational power in combination with the new API's gives us the motivation to experiment and expand the research frontiers.

1.3. Literature related to the Thesis methodology and objectives.

Di persio and Honchar (2017) Use an LSTM network, a basic multi-layered Recurrent Neural Network and a Gated Recurrent Unit to predict the stock price of Alphabet Inc. They implement different time horizons for the purpose to explain the related different dynamics. They argue that their approach makes it possible to

handle long sequences, especially in the case of LSTM. Additionally, the performances tend to be very high despite the use of different time horizons. The accuracy they claim reaches up to 72%.

Nelson, Pereira and Oliveira (2017) use LSTM networks to predict the trends of stock markets based on historical prices in comparison with technical analysis. To achieve that they construct a forecasting algorithm and experiment analyzing results using a number of metrics. They measure whether the algorithm has a better performance compared to other Artificial Intelligence methods and strategies.They claim very promising results reaching up to an average of 55,9% accuracy on stock predictions.

Koyabashi and Shirayama (2017) present a new methodology for predicting time series involving many deep learning algorithms in combination with a Bayesian network. They implement three types of deep algorithms: A LSTM network, an RNN and a deep Multilayered perceptron. A naïve Bayes classifier then decides the deep learning algorithm that is in charge for every time series. They apply their methodology to the Nikkei Average Stock price dataset to measure the accuracy of their predictions. They demonstrate how their methodology improves the F-value and the accuracy compared to using a single deep learning algorithm.

Fischer and Krauss (2017) use LSTM neural networks to forecast the direction of the movement of the constituent stocks of S&P 500 from 1992 to 2015. Their LSTM network achieved returns up to 0.46% with a 5.8 Sharpe Ratio before transaction costs. Consequently, it

outperformed memory free classifiers like random forests, deep neural networks and logistic regressions. They found that the selected stocks have as common attributes their high volatility and a short-term reversal return profile. Exploiting these findings, they argue that they are able to formulate a rule-based a short-term reverse strategy that is responsible for a portion of LSTM returns.

Di Persio and Honchar (2016) experiment with different Neural Network architectures and analyze the nature of financial time series. After making a literature review on the topic they present the Multi=Layer Perceptron, the Convolutional Neural Network and the Long Short-Term Memory recurrent neural network types. They argue that the choice of the correct variables and the data preparation for every particular type of machine learning technique is very crucial to the accuracy of the results. They experiment on the S&P 500 historical time series, making forecasts about its trends using past values. They propose an original approach using convolutional neural networks and wavelets in order to outperform the traditional Neural Network Architectures. Their results present that Artificial Neural Networks are capable to predict financial trends even when they are trained on plain past values time-series data and suggest more ways to develop further research.

Ghiasssi, Saidane and Zimbra (2005) Compare a number of novel and traditional quantitative forecasting methods using standard benchmarks for time series research. They find that feed forward neural networks and their own architecture of Dynamic Artificial Neural Network perform better than the traditional ARIMA techniques.

1.4. Thesis outline.

In the following parts of the Thesis, we cover consecutively the financial theories about Efficient Market Hypothesis, asset pricing and forecasting models. Following is the mathematical background that supports our research questions. Next is the part that we present the methodology, the data and the results. Further down we discuss the results and we make our final remarks.

Analytically in part two, we describe the statistical properties of assets returns (Cont, 2000) and efficient market hypothesis (Fama, 1970). We also make a brief presentation of the main academic and practical approaches that challenge the Efficient Market Hypothesis in the modern financial world like technical analysis, behavioral finance, value investment principles and the CAPM model. We also discuss how the Futures prices of indices are derived and the macroeconomic variables used to predict market behavior.

In the third part, we discuss the universal approximation theorem (Cybenko,1989) and the math behind several types of Artificial Neural Networks and their limitations. We describe analytically how the algorithm of a Multilayered perceptron operates in algebraic level and we also explain the comparison models we use.

In the fourth part, we thoroughly present the models we will use and the input data. We also present the evaluation methods of our models and the results of the back-testing strategy for each one of them. Finally, in the parts five and six, we discuss our results in comparison with our research questions and we conclude on the findings of the Thesis.

Chapter 2. Literature Review.

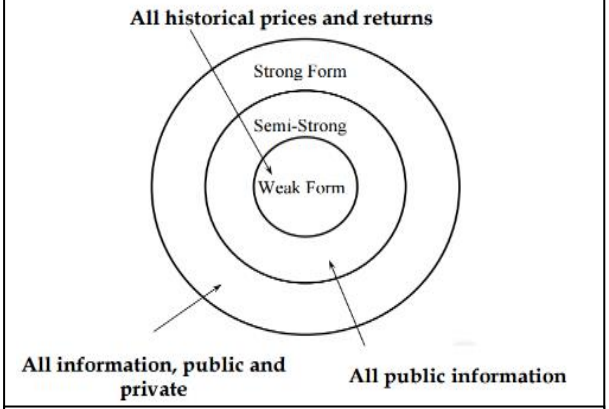
In this part, we will present the general financial theoretical literature that relates to our research interests. We will first attempt to present the main points of the Efficient Market Hypothesis. Then we will present some of the dominant theories and practices that attempt to support the active investing framework. Finally, we will try to clarify the basic statistical attributes of our input data.

2.1 Efficient Market Hypothesis.

Papers relating to the Efficiency of Markets go back to the sixtieth century (Sewell, 2011). Gerolamo Cardano writes in his book "*Liber de Ludo Aleae*" -The Book of Games of Chance- (Cardano,1564) that participating in a zero-sum game which cannot be modeled by using its variables makes you a fool or a crook.

Bachelier (1900) approaches the concept of Efficient Market describing it as a "fair game". His model was lacking the risk factor thus it was failing to provide a working application of his theory. In his famous Efficient Market Hypothesis paper, Fama (1970) presents his view on the self-adjusting mechanics of the market and argues that all available information is

Figure 1. The Efficient Market Hypothesis



reflected in every current price. He supports that the deviations from efficient values are purely random and that no method is capable of a consistent forecast. Subsequently, he describes EMH as having three forms:

- a weak form testing how well do past prices returns predict future returns.
- a semi-strong form testing how quickly security prices reflect public information and
- a strong form that tests if there is any confidential information not reflected in the market prices (Fama, 1991).

Fama's theory implies that the best strategy for an investor is the passive one. This consists of buying and holding a percentage of every asset in the market in portions that are pre-weighted by the total market capital allocation or follow the allocation of an index. In this rationale, the investor must follow the market as accurately as possible (Laopodis, 2012).

A big contradiction is that market efficiency theory holds structurally upon the presumption that market inefficiencies are exploited by the investors fast and massively enough that the prices return to their efficient value almost immediately. This is known as the market efficiency paradox. (Gilson, R. J. and Reinier H. 1984)

Since then a lot of research has been done testing all three forms of market efficiency using various analytical tools. Stout (2003) Mentions that Fama's paper about market efficiency has already three decades impacting greatly the financial world. He argues that during that time we have observed a series of empirical evidence that challenges it. Pricing anomalies, demand

inelasticity, excessive volatility, delayed information response and consistently superior traders are some examples of them. He argues that novel ways of thinking about market dynamics are already developed and states the heterogeneous expectations asset pricing models, the research about the limits of arbitrage and the incorporation of novel types of information into prices and the growing literature on behavioral finance. In this Thesis, we are going to test the weak form of the EMH.

2.2 Behavioral Finance

Efficient Market Hypothesis rational investor theory has been challenged the past years due to the emerging ideas based on the field of behavioral finance (Kostandinidis et al 2012).

This discipline supports that market inefficiencies have their causality in social, emotional and personal sources (Ricciardi and Simon, 2000). The fundamental principle of behavioral finance is that perception is subjective thus every person has a different way of internalizing financial indicators (Garcia, 2013). Garcia (2013) presents a number of psychological biases, namely cognitive dissonance, disjunction effect, the illusion of knowledge, overconfidence bias and Loss aversion.

Analytically, Cognitive dissonance describes the bias effect that occurs when a person disagrees with a statement and he tends to ignore the information that is in line with that statement as well. The opposite may occur for the statements he agrees with.

Disjunction effect is the bias where a person ignores or supports a set of information according to prior knowledge of an occurring

event. The illusion of knowledge is a behavioral bias where an individual believes he can make better decisions because he has more information than the average person.

Overconfidence bias is the bias where an experienced investor ignores market signals and rational consequences because he thinks he has superior skills compared to the average investor.

Loss aversion is the bias where the investor is afraid to lose his gains. Consequently, he sells immediately. This pushes the price of good securities down. (Perren et al, 2015).

Herding is the bias that describes the behavior of investors who due to lack of confidence or financial education mimic the investment strategies of others (Basu et al, 2008). This is also called "monkey trading". Basu et al (2008) present in their study that during the period 1993-2003 a lot of herding and overconfidence supported trades affected the NASDAQ and S&P 500 indices. The study concluded that investors were not acting rationally and supported the behavioral finance approach.

De Bondt and Thaler (1985) State that empirical evidence based on New York Stock Exchange monthly returns data reveal that investors overreact in unexpected and dramatic news events in a way that produces substantial weak form market inefficiencies. Thirty-six months after portfolio formation the "losers" stocks are found to outperform the "winners" stocks by 25%. They mention that recent research in experimental psychology also supports this hypothesis in violation of Bayes rule. Additionally, they argue that according to Kahneman and Tversky (1973):" the predicted value is selected so the standing of the case in the distribution of outcomes matches its standing in the distribution of impressions".

This is an instance of what Kahneman and Tversky call the representativeness heuristic. In another research paper, Shankar and Dhankar(2015) supported the need for a theory that explains the gap that is observed between academic financial theories and actual trading economics. They state that psychology can contribute a lot to the enhancement of the classical economic theory.

2.3 Asset Pricing Models

Asset pricing models are developed in order to predict the future values of different kind of assets which market price depends on the relationship between several variables. We will present the main asset pricing models namely the Capital Asset Pricing Model, Arbitrage Pricing Theory, the value investor fundamentals approach and finally the technical analysis method.

2.3.1 Capital Asset Pricing Model

Markowitz Portfolio Theory set the rules to tune our asset allocation in a way that we can achieve higher returns with the lowest possible risk (Markowitz, 1952). The theory assumes that our portfolio is fully diversified in a way that non-systemic risk is absorbed (Reilly and Brown, 2012). This opened the road to the formulation of the Capital Asset Pricing Model (CAPM) which incorporates the non-systemic risk and can be used to evaluate portfolios that are not completely diversified. According to CAPM the portfolio can still be optimized by diversification and exposure to non-systemic risk can mitigate (Hodnett and Hsieh, 2012).

Capital asset pricing model variables are the risk-free rate usually expressed by the 10 years government bond returns, the non-systemic risk measured by beta and the expected market returns expressed usually by the returns of a major market index.

The CAPM formula:

$$E(Ri) = RFR + \beta i[E(Rm) - RFR] \quad (1)$$

Where: $E(Ri)$ is the expected return,

RFR is the risk free rate,

βi is the beta of the instrument,

and $E(Rm)$ is the expected return of the market,

(Reilly and Brown, 2012*)*

We can observe how beta coefficient accounts for the non-systemic risk by being multiplied by the difference of the returns of the market to the risk-free rate (Perolds, 2004).

The capital asset pricing model is widely accepted by the financial world due to the fact that is very easy and straightforward (Fama and French, 2004). Several researchers like Mullins (1982), support that the CAPM is an oversimplification of the market mechanics with its variables being very volatile and the beta being subjective to time periods and which index is used as a reference. Fama and French (2004) state that despite other models being developed since the CAPM model remains the main reference of asset pricing in academic literature.

2.3.2 Arbitrage Pricing Theory

According to Ross (1976), a better asset pricing model is the Arbitrage Pricing Theory (APT). The APT is far less restrictive and provides a more generalized approach to the use of beta coefficient than the identical "market portfolio" used by the CAPM. The APT uses a series of systematic risk factors to find the "true" market price of an asset in order to identify over or undervalued securities and allow the investor to order accordingly before the prices are corrected by the market (Hodnett and Hsieh, 2012).

The APT formula is:

(2)

$$r_j = a_j + b_{j1}F_1 + b_{j2}F_2 + \cdots + b_{jn}F_1 + e_j$$

Where: a_j is a constant for asset J,

F_1 is a systematic factor,

b_{j1} is the sensitivity of the jth asset to factor k,

and e_j is the risky asset's idiosyncratic random shock

with mean zero.

The Arbitrage pricing model has the downside of not accounting for the systemic risk factors that affect returns such as inflation, industrial production, interest rates etc. (Hodnett and Hsieh,2012). It is apparent that theBehavioral finance poses a challenge for Arbitrage Pricing Theory in the context that stock returns are also affected by psychological and other non-quantitative factors. (Galagedera, 2007).

2.3.3 The Value Investor approach

David Dodd and Benjamin Graham set the principles of what is called Value Investing. Both Dodd and Graham where professors at Columbia University and in 1934 they co-published the book "Security Analysis" that set the foundations of the Value Investing method of selecting mispriced assets. Later Benjamin Graham wrote his famous book "The Intelligent Investor" in which he provides a practical approach to recognize undervalued companies.

According to Graham (1973) the general criteria to identify companies that are likely to have shares producing abnormal returns are at a glance:

- A company must have an earning to price yield at least double the Risk-Free Rate.
- 2) The companies Price to Earnings Ratio must less than 40% of its historical high for the past 5 years.
- 3) The company's stock dividend yield should be at least two-thirds of the Risk-Free Rate.
- 4) The company's stock price should be below two-thirds of the company's book value per share.
- 5) The stock price should be below the companies two-thirds of its Net Current Asset Value.
- 6) The company's total liabilities should be less than the company's book value.

- 7) The company's current ratio should be over 2. 8) The total liabilities of the company should be smaller than 2 times the Net Current Asset Value.

- 9) The earnings growth of the company for the past 10 years should consist of 7% compound rate.

- 10) Earnings growth should not decline more than 5% for the past 10 years. Benjamin Grahams student Warren Buffet wrote for Graham that he has "never met someone with a mind of similar scope" (Buffet, 1976).

Buffet himself is appraised as the most successful value investor of all times. Additionally, many other papers have been published about Graham's value investing approach due to the fact that it is so much supported by practitioners and has been proved to create abnormal returns in the past fifty years.

A study by Oppenheimer and Schiarbaum (1981) uses Graham's investment criteria to test the efficient market hypothesis. They concluded that value investment strategies can produce alpha in a market that was virtually semi-strong efficient.

Another study by Oppenheimer and Schiarbaum (1983) uses value investment strategies to test whether it can produce alpha for property-liability insurers and pension plans. They argue that Graham's strategy produced alpha and that it is a more efficient strategy than the ones used by most Hedge Funds either passive or active. In 1984 Oppenheimer presented another study proving that portfolios allocated using the value investment criteria outperformed the market indices for the case of New York Stock Exchange (Oppenheimer, 1984).

Ye (2013) examined the returns of a portfolio created following the value investment method and consisting of stocks traded on Shanghai stock exchange. He concluded that Grahams method

outperformed the Shanghai stock index for most of the years between 2006 and 2011.

In a relevant paper about the case of FTSE Bursa Malaysia EMAS index Chang (2011) used Graham portfolio criteria to create a portfolio and compare its performance with the index. He found that for the years 2000 to 2009 the value investment portfolio outperformed the index for each one of the years studied.

Finally, Singh and Kaur (2014) tested value investment strategy proposed by Graham for the Indian stock market. Their findings show that the companies who fulfill at least five of Graham's criteria had significant excess returns compared to other companies. Regression analysis showed that not all of the criteria were applicable to companies trading at the Indian Stock Market. The study concludes that variables affecting alpha formation the most are: earnings Yield, discount to tangible book value and net current asset value combined with low leverage and stability in earnings.

2.3.4 The Technical Analysis Approach

Technical analysis assumes that all information that is publicly or privately available about an asset is compiled in its market price. The methodology uses indicators to identify and exploit long lasting trading patterns of behavioral nature (Brown and Jennings, 1989).

Levy and Post (2005) state that technical analysis is based on the assumption that markets always follow a cycle consisted of rising and falling trends that can be broken down to short, intermediate and long-term intervals. Analysts then use charts to identify and

exploit the trends using moving average and strength indicators. Technical indicators can be divided into breadth indicators which try to identify markets strength and weaknesses and sentiment indicators which are used to measure how other investors are feeling. As an example of a sentiment indicator Pring (2014) mentions the VIX which is the Chicago Board Options Exchange Market Volatility index and it is considered by technical analysts to be the measure of "fear" in the market.

Technical analysis was overlooked by the early academic community literature based on the fact that is not in accordance with the highly appraised Efficient Market Hypothesis (Kavajecz and Orders, 2004). Fama and Blume (1966) reject technical analysis methods by empirical experiments that attempt to prove that the method is not capable of forecasting future values.

Nevertheless, more recent literature is presenting evidence that technical indicators can reveal inconsistencies between spot prices and true values thus being able to predict market reactions (Neftci and Policano,1984).

More specifically Academic research on the performance of Technical indicators for the New York stock exchange has been conducted by Brock et al. (1992). They used two technical indicators namely, a moving average oscillator and a trading range breakout on a data set made out of Dow Jones Industrial Average from eighteen-nighty-seven to nineteen-eighty-six. They claim that their results provide significant evidence that technical strategies can produce abnormal returns on the American stock exchange. Similar studies have been conducted to test Technical Analysis performance for the futures markets.

Lucac and Brorsen(1990) used twenty-three different technical strategies on thirty futures markets for the decade between nineteen-seventy-five and nineteen-eighty-six. They found that only seven out of twenty-three technical strategies generated gains on a monthly basis after transaction cost.

Considering the Forex market, a study by Qi and Wu (2006) testing technical indicators on several currencies worldwide supports that it can generate abnormal returns even after transaction costs. They claim that alpha reached up to 12 percent for all major world currencies with an exception for the Canadian dollar for which no technical strategy manages to generate positive returns.

2.3.5 Macroeconomic determinants of stock market returns

In this section, we will present the macroeconomic variables that may affect stock markets according to academic literature.

Studying the case of the Greek stock exchange, Patra and Poshackwale (2006) found that exchange rates had no positive or negative correlation with stock market performance where money supply, trading volume and consumer price index were found to affect Athens Stock Exchange performance both in the long and short-term. They also conclude that the Greek stock market was semi-strong inefficient as publicly available information on macroeconomic variables could be used to predict future prices.

Narayan et al. (2014) investigate the macroeconomic determinants of stock prices using data from 13 Indian commercial banks. They use the methodology of panel data. They conclude that exchange rates and industrial production are

positively correlated with stock markets. Inversely interest rate has a negative correlation with stock markets. All macroeconomic variables examined were found to be statistically significant in a long-term period, with industrial production to be more effective in comparison with the other variables.

The case of Pakistan was studied by Sohail and Hussain (2009). Macroeconomic variables where examined for the period of 2002 to 2009 and found to be positively correlated with stock returns. In a contradictory research Kutty (2010) found that there is no causality between exchange rates and stock market returns although it appeared to have a short-term correlation.

The case of South Africa was investigated by Ali et al (2016) where it was found that from 1998 to 2010 the macroeconomic variables of: inflation, industrial production, exchange rates and money supply had long-term impact on stock market returns. The effect of industrial production was found to be more intense than the other three variables.

Macroeconomic variables of Japan seemed to have no significant impact on stock movement according to Kurihara and Nezu (2010). For the case of Japan, the main determinant that influences the market seemed to be the movement of the United States stock market. This relevance was tested and found to be long term. The general picture of the literature reveals that the main macroeconomic variables that affect stock prices are exchange rates, industrial production and money supply.

2.4 General Statistical Properties of financial returns data

Modeling with Neural Networks is advantageous due to the fact that you don't need to make any a priori assumptions about the functional form or the distribution of the data. Nevertheless, financial returns datasets have some general statistical properties as a time series. Cont (2001) summarizes these properties into nine as follows:

i. No linear autocorrelation.

ii. The slow decay of linear autocorrelation of absolute returns.

iii. Heavy tails.

iv. Larger tail for losses than for positive returns.

v. The shape of the distribution changes through time.

vi. Volatility clustering.

vii. Conditional heavy-tails (even after adjusting for volatility clustering).

viii. Leverage effect. Volatility is negatively correlated with returns

ix. Correlation between trading volume and volatility.

Cont (2001) concludes that the above statistical facts are the result of generalized assumptions of qualitative nature and are not constraints that relate to a specific model. He further questions whether these facts are restrictive to certain modeling methodologies used in economic research.

2.5. Determination of Futures Prices for Stock Indices.

Since we will be using the futures prices of the Athens Stock Exchange index we will try to present how we derive the future price from the index value. Hull (2008) states that since an index could be viewed as a portfolio of dividend-paying assets, the futures price of an index could be calculated by equation as

$$F_0 = S_0 e^{(r-q)T} \quad (3)$$

Where q is the dividend yield rate of the index,

F_0 is the futures price today,

S_0 is the underlying index spot price,

r is the free interest rate,

and T is the time interval of the contract

(Hull, 2008)

There are empirical results that show that the influence between the futures price and index values are many times bidirectional with many examples of the futures instruments taking the lead.

Stoll and Whaley (1990) research the dynamics between stock index futures and stock indices returns. They find that in the case of S&P 500 the returns of the futures and the stock market are for the most part "contemporaneous". Nevertheless, they observed evidence that the futures market leads the stock market due to the fact that the futures market index is not subject to trading hours.

2.6 Artificial neural networks.

Neural Networks are inspired by their biological counterparts and they can have different attributes, architectures and sizes. They operate in resemblance to the human brain neural structure. This practically means that they are trained by the incoming data to develop the best algorithm that maps every possible subset of the data to the future values, so it can find solutions to address various future situations. After going through multiple trial and error adjustments it ends up with the optimal weight matrix to use for solving future challenges (Shapiro, 2000).

The idea of an artificial neural network is born in the first half of the twentieth century when McCulloch and Pitts (1943) wrote the first article discussing the properties of an ANN named "*A logical calculus of the Ideas Immanent in Nervous Activity*". The first presentation of a single layer perceptron is in the book "*Principles of Neurodynamics: Perceptrons and the Theory of brain*

Mechanisms" by Frank Rosenblatt (1962). Rosenblatt describes a feed-forward neural network with activation functions that "fire" over a threshold. His paper was very influential in the sciences of biology and mathematics but the lack of computational resources at the time made further research very difficult.

A huge breakthrough was made by George Cybenko in 1989 in his paper "*Approximation by superpositions of a sigmoidal function*" where he proves the "universal approximation theorem" which states that any feed-forward network using sigmoidal functions as activation neurons can successfully approximate any continuous function given the appropriate inputs. In 1991 the theorem was generalized further by Kurt Hornik in his paper "Approximation Capabilities of Multilayer Feedforward Networks" where he proved that the architecture is the crucial element of the ANN that gives it the attribute of universal approximator and not the type of the activation function it uses.

Today Artificial Neural Networks have developed tremendously and are used casually in the field of machine learning and Artificial intelligence. They are the core of applications like Google translate and self-driving vehicles. Google has also developed the dominant open source Application Programming Interface (API) for ANN's at the moment called "*Tensor flow*" (Goldsborough, 2016). In this Thesis, we will suggest the use of an ANN to test the efficient market hypothesis and present relevant literature and empirical review on the field.

2.7. Neural networks within finance.

In this part, we will present a selection of papers that provide empirical evidence of the use of Artificial Intelligence technology in financial applications. The literature researches the performance of various A.I types as a predictor of market movement and in several cases, compares it with classic econometric models and technical indicators.

The main reasons for the outburst of studies around A.I for finance the recent years are the abundance of data in combination with the advancements in processor technology and cloud computing that led to an unpreceded offer of computational power. Deep Learning algorithms are used today by big financial firms. Although financial institutions don't make their outcomes and procedures publicly available, open crowdfunded platforms about machine learning for financial forecasting exist online like Quantopian (Quantopian, 2017) and Numerai (Numerai, 2017).

Artificial neural networks (ANN) is a very successful example of machine learning forecasting technique since it is data-driven, self-adaptive and does not need any statistical presumptions about the data (Lu, Lee and Chiu, 2009). In the last decades, ANN's have become very popular in the field of financial market forecasting. What gives the ANN the competitive advantage in these type of problems is that these mechanisms are able to handle data characterized by nonlinearities, discontinuities and high-frequency polynomial components (Liu & Wang, 2012).

Artificial Neural networks are used for financial forecasting for over twenty years now (Faddala and Lin, 2001). Although they were usually single layered perceptrons they outperformed

econometrical and statistical models mainly due to the nonlinearities and complex structure of the inputs used in financial variables. Several examples of the use of ANN's include financial instruments market price forecasting, portfolio management algorithms, fraud detection and default risk evaluation (Johnson and Winston 1994).

The following papers are a brief presentation of research involving neural network technology in the financial field.

Tsaih, Hsu and Lai (1998) try to predict the daily price changes of S&P 500 stock index futures using a hybrid technique of Artificial Neural Networks and rule-based systems. They argue that the hybrid approach can more reliably model and support the decision-making processes. They claim that their results show that their strategy outperformed the passive buy and hold investment strategy during a 6-year testing from 1988 to 1993.

Zhang and Wu (2009) Try an optimized backpropagation neural network model in order to predict the S&P 500 movement. They use an algorithm inspired by bacterial foraging behavior to train the artificial neural network and update its weights. Their sample is 2350 trading days and consists of the closing price, the opening price, the lowest price, the highest price, the total volume of stocks traded and the adjacent price. They argue that the enchased model is less complex to compute, has better prediction accuracy and needs less training time compared to the classic backpropagation model.

Wu, Luo, Li, Wang and Chen (2015) present a new method that combines a set of Artificial intelligence prediction methods in a series meaning that the first model's output is the following models

input. They claim that the existing single or fusion models cannot deal with complex situations. In contrary, their model gives operation suggestions by getting results from quantitative methods and using qualitative methods to decide the operation's strength. They conclude that their hybrid model in comparison with the sole models gain the highest overall accuracy and reach a value up to 75.3%.

Hilovska and Koncz (2012) briefly examine most contemporary Artificial Intelligence applications for financial markets and various techniques. They focus on Data Mining, Expert Systems, Artificial Neural Networks, Genetic Algorithms, Fuzzy Systems and Multi-Agent Systems. They conclude as of which application and technology they examine contributes more in optimizing market efficiency.

Bahrammirzaee (2010) investigates the use of artificial intelligence methods to assess highly nonlinear, time-variant problems that present a significant level of complexity. He finds that the accuracy of these methods can be superior to that of traditional statistical ones however this superiority is not absolute. He more specifically examines the Artificial Neural Network applications, the Expert System applications and the Hybrid Intelligent Systems applications.

Armano, Marchesi and Murru (2005) present a hybrid genetic-neural architecture. They feed the Genetic part with technical analysis data and the neural part with past stock prices. They test their forecasting system in two stock market indexes. The results are reported to repeatedly outperform the "Buy and Hold" strategy.

Boyd and Kaastra (1996) try to provide a practical guide to design a neural network forecasting model into eight steps and identify the success factors of their methodology namely: patience and resources, the programming mainframe, a good historical record of what works and what not.

Charef and Ayachi (2016) compare an Artificial Neural Network to a Generalized Autoregressive Conditional Heteroskedasticity model for exchange rate forecasting. They found that the Artificial Neural Network remains undoubtedly the most efficient model compared to traditional statistic models. Enke and Thawornwong (2005) Examine Neural Network models for their ability to provide an effective forecast of futures values. They find that the trading strategies guided by the classification models generate higher-adjusted profits than the buy and hold strategy.

Guresen, Kayakutlu and Daim (2011) use a multi-layer perceptron, a dynamic artificial neural network (DAN2) and hybrid neural networks which use generalized autoregressive conditional heteroscedasticity (GARCH) to extract new input variables. The comparison for each model is done in two viewpoints: Mean Square Error (MSE) and Mean Absolute Deviate(MAD) using real exchange daily rate values of NASDAQ Stock Exchange index. The results showed that classical Artificial Neural Network Multilayered Perceptron outperformed the dynamic artificial neural network and the hybrid neural networks with GARCH.

Habib and Hamid (2014) present a comprehensive survey of the application of Artificial Neural Networks in various areas of finance and economics. Then they present the steps of using neural networks to forecast the volatility of the S&P 500 index futures

prices. They find that Neural Networks forecasts outperform implied volatility forecasts and they approach significantly realized volatility.

Kuo, Chen and Hwang (2001) develop a genetic algorithm based fuzzy neural network to measure the qualitative effect on the stock market then they integrate it with technical indexes through an artificial neural network. Evaluation results indicate that the neural network considering both the quantitative and qualitative factors outperforms the neural network considering only qualitative factors.

Wang, Wang and Zhang (2011) propose a novel approach to forecasting via the Wave-let De-noising-based Backpropagation neural network and compare it with a single Backpropagation neural network. They find that the data denoising procedure used by the Wave-let De-noising-based Back Propagation neural network improved significantly the forecasting accuracy compared to the neural network using raw data.

Wei, Kin, Nakamori, Shouyang and Lean (2007) discuss the input variables and type of neural network models for the prediction of Forex, stock market index and economic growth. They conclude that most neural network inputs for exchange rate prediction are univariate, while those for stock market index prices and economic growth predictions are multivariate. They also remark that the performance of neural networks is improving when they integrate with other technologies.

Zhang, Patuwo and Hu (1998) present a state-of-the-art survey of artificial neural networks application in forecasting and review their advantages and their limitations.

Theofilatos, Likothanasis and Karathanasopoulos (2012) compare five state-of-the-art machine learning techniques in trading the EUR/USD exchange rate. They use supervised classification with K-Nearest Neighbors algorithm, Naïve Bayesian classifier, Artificial Neural Networks, Support Vector Machines and Random Forests. For comparison reasons, they benchmark the techniques by two "traditional" techniques namely a "naïve" strategy and a moving average model. The results showed that the machine learning techniques outperformed the traditional techniques in terms of annualized returns and Sharpe ratio.

Chapter 3. Methodology

We will start building up the methodology of the Thesis by introducing the mathematical concepts around Artificial Neural Networks. We will begin our presentation with a concise introduction to the "universal approximation theorem". Then we will proceed with a descriptive analysis of the various Artificial Neural Networks types and their training procedures. In the subsections to follow we will briefly comment the concept of Dropout Regularization and the Adaptive Moments Estimator algorithm . We will also present the algebraic expressions of the comparative methodologies. Finally, we will present in detail the application of our methodology and justify the experiments that we will use to address our research questions.

3.1 Universal approximation theorem

The universal approximation theorem supports that "a single hidden layer feedforward network with an arbitrary number of hidden nodes can approximate any given continuous function"(Cybenko,1989). Cybenko's proof of the theory presumes that the only type of function used in the ANN is a "sigmoid" function.

Kurt Hornik proved the theorem for an arbitrary continuous bounded function in 1991 given that there is an adequate number of neurons in the hidden layer (Hornik,1991).

The latest developments in the topic where presented by Sonoda and Murata (2017) who substantiated that an unbounded function can be used and paved the way for the "Rectified Linear Unit function" (ReLU) to be the new state of the art function for Deep Learning.

3.1.1. Universal approximation theorem algebraic expression:

Let $\varphi(\cdot)$ be a bounded, and monotone-increasing continuous function. Let ID denote the D-dimensional unit hypercube $[0,1]^{\mathrm{D}}$. The space of continuous functions on $\mathrm{I_D}$ is denoted by $C(I_D)$. Then, given any function $f \in C(I_D) and\ \varepsilon > 0$, there exist an integer M and sets of real constants, a_j, b_j, w_{ij}, where $j = 1, \dots, M$ and $i = 1, \dots, D$ such that we may define:

(4)

$$F(x_1, \dots, x_D) = \sum_{j=1}^{M} a_j \varphi \left(\sum_{i=1}^{D} w_{ij} x_i + b_j \right)$$

as an approximate realization of the function $f(\cdot)$; that is,

(5)

$$|F(x_1 \dots x_D) - f(x_1, \dots, x_D) - f(x_1, \dots, x_D)| < \varepsilon$$

for all $x_1, x_2, \dots, x_D$ that lie in the input space.

3.2. The Single-Layer Perceptron.

According to Bishop (1995), We can denote a generalized formula of a single-layer feed-forward neural network as follows:

(6)

$$y_k = g\left(\sum_{t=0}^{D} w_i x_i \right)$$

Where: y_k is the output,

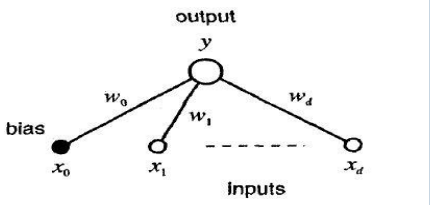
$g(\cdot)$ is an activation function.

x_i is input and w_i is its corresponding weights.

$x_0 = 1$

w_0 is a bias parameter.

Figure 2. The Single Layered Network



Source: (Bishop,1995)

The biases nodes weights are adapting sharing the same procedure with all other ANN nodes weights. It is also implied that the activation function $g(\cdot)$ functional form defines the character of the ANN. Elaborating on that concept we can say that if $g(\cdot)$ takes the form of $g(x) = x$ then the ANN in formula (6) takes the form of a linear regression. Consequently $g(\cdot)$ taking the form of a piecewise function of the type:

(7)

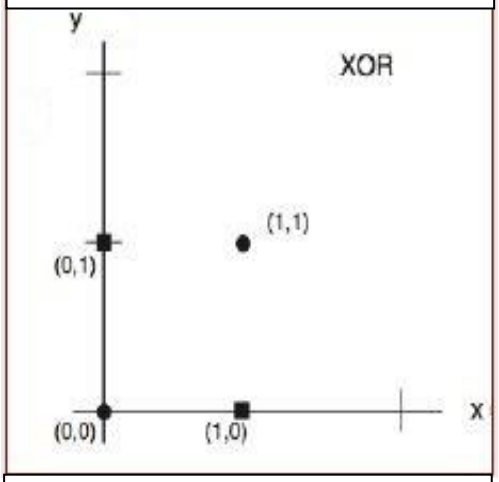
$$g(a) = \begin{cases} 1 \; if \; a > 0 \\ \\ 0 \; otherwise \end{cases}$$

turns the ANN into a binary classifier and in the case, that $g(\cdot)$ is a sigmoid function we have to do with an ANN which takes the form of a logistic regression. Therefore, we can observe how the wrapping function of an Artificial Neural Network node can define its mechanics and this gives us an insight of its generalization capabilities.

3.2.1. Limitations of single layer networks.

According to Minsky and Papert (1969),a single-layered perceptron is not able to approximate a classifying function for a data set that is not linearly separable. This is demonstrated by training an NN to learn the XOR logical operation where $(0,1) \wedge (1,0) \in \mathbb{R}^2$ form class A and $(0,0) \wedge (1,1) \in \mathbb{R}^2$ form class B. The NN fails to find a separator for classes A and B.

Figure 3. The XOR limitation



Source: (Minsky and Papert,1969)

Additional limitations for the single-layered NN can be noticed in regressions where the linearity of the activation function doesn't allow good fitness of the model on the data. Thus, single layered architectures are used today for academic purposes rather than real-world applications.

3.3 The multilayered perceptron.

The limitations of the single-layered neural network demonstrated were decisive for the dereliction in Neural Network research during the 1970s. The discovery of the Multilayered Neural Network trained by backpropagation of errors emerged as a tool for solving a variety of problems (Rumelhart, et al. 1986).

As illustrated in figure 4. a Multi-Layer Perceptron has a minimum of three layers of nodes (neurons): The input layer (X), the "hidden" layer (Z) and the output layer (y). Observing formula (8)

we can remark that it practically equals the Universal Approximation Theorem formula (4) in section 3.1 as long as the activation function stays linear (Haykin, 1998). What is important to notice is that the output of the first layer is again multiplied by a second weight matrix and fed to a new layer of nodes. Considering that the outputs of the layers are non-linear we can imagine the level of complexity built over time.

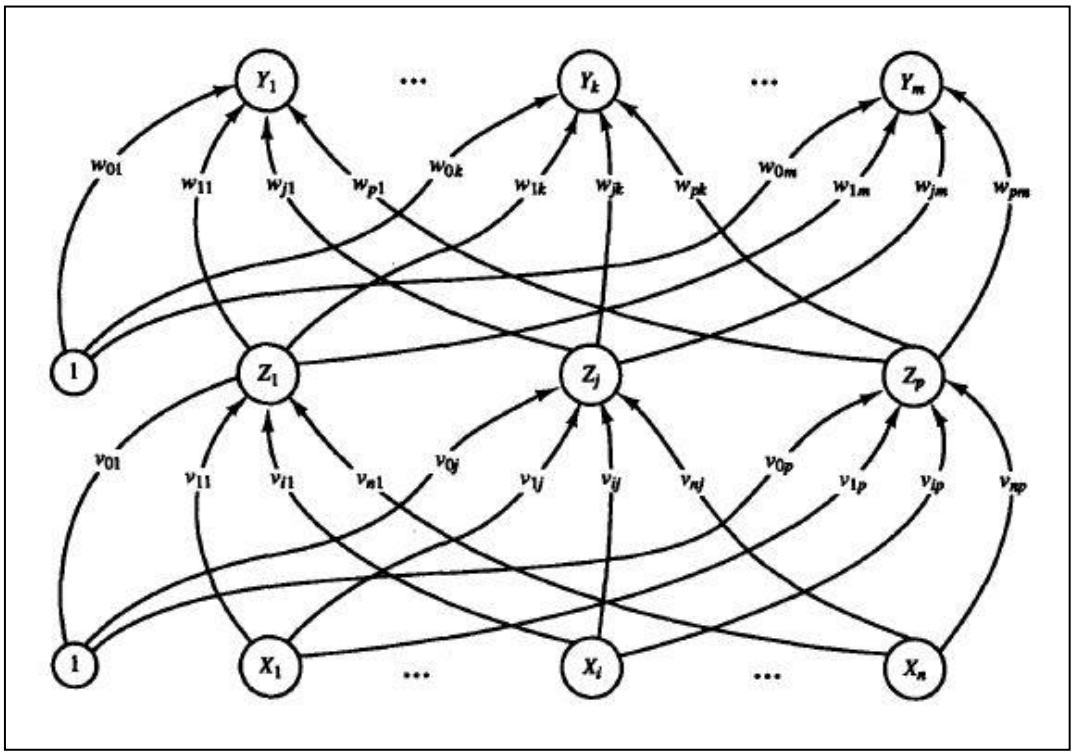
(8)

$$y_k = h\left(\sum_{j=0}^{M} w_{kj} g(a_j)\right)$$

(9)

Where $a_j = \sum_{j=0}^{D} w_{ij} x_i$

Figure 4. The Multi-Layer Perceptron



Source Fausett (1994)

Where:

x is the input training vector $x = (x1, \ldots . . xn)$

t is the output target vector $t = (t1, \ldots . . tn)$

X_i is the input unit i.

Z_j is the hidden unit j.

The net input to Z_j is denoted $znet_j$.

Y_k is the output unit k.

The net input to Y_k is denoted $ynet_k$.

W is the weight matrix between Z and Y .

V is the weight matrix between X and Z .

δ_k is the error correction weight adjustment for wjk.

δ_j is the error correction weight adjustment for vij .

a is the learning rate.

v_{0j} is the bias weight on the hidden unit.

w_{0k} is the bias weight on the output unit.

3.3.1 The Multilayered Perceptron Training Algorithm.

The perceptron processing pattern is as follows:

The values of the explanatory variables pass through the input nodes. The hidden layer nodes pass all incoming information through an activation function. If the value calculated by the activation function is above a certain threshold it proceeds to the output layer (Dunis et al,2012).

Each node of a layer has connections (W, V) to all the nodes of the next layer that are weighted. We train the perceptron by adjusting the weights so that the artificial neural network can map the values of the training data to the relevant output values. The training starts with random weights and continues applying a supervised learning algorithm, called backpropagation of errors. (Shapiro,2000)

Fausett (1994) describes the whole ANN operation algorithm as follows:

Step 0: Initialize weights. (Set to small random values).

Step 1: While stopping condition is false, do Steps 2-9.

Step 2: For each training pair, do Steps 3-8.

- ***Feedforward:***

Step 3: Each input unit $(X_i, i = 1, \dots, n)$ receives input signal Xi and broadcasts this signal to all units in the next layer (the hidden units).

Step 4: Each hidden unit $(Z_j, j = 1, \dots, p)$ sums its weighted input signals,

(10)
$$znet_j = v_{0j} + \sum_{j=1}^{n} x_i \; v_{ij}$$

applies its activation function to compute its output signal,

(11)
$$z_j = f(znet_j)$$

and sends this signal to all units in the next layer (output units).

Step 5: Each output unit $(Yk, k = 1, \dots, m)$ sums its weighted input signals,

(12)
$$ynet_k = w_{0k} + \sum_{j=1}^{p} z_j \; w_{jk}$$

and applies its activation function to compute its output signal,

(13) $$y_k = f(ynet_k)$$

- ***Backpropagation of error:***

The backpropagation algorithm compares the output with the optimal value and calculates the gradient descent of the error function.

Step 6: Each output unit $(Y_k, k = 1, \dots, m)$ receives a target pattern corresponding to the input training pattern, computes its error information term,

(14)
$$\delta_\kappa = (t_k - y_k) f'\ (ynet_k)$$

calculates its weight correction term (used to update W_{jk} later),

(15)
$$\Delta w_{jk} = \alpha \delta_\kappa z_j$$

calculates its bias correction term (used to update W_{0k} later),

(16)
$$\Delta w_{0k} = \alpha \delta_k$$

Step 7: and sends δ_k to units in the previous layer. Each hidden unit $(Z_j, j = 1, \dots, p)$ sums its delta inputs (from units in the next layer),

(17)
$$\delta net_j = \sum_{k=1}^{m} \delta_\kappa w_{jk}$$

multiplies by the derivative of its activation function to calculate its error information term,

(18)
$$\delta_j = \delta net_j \, f'(znet_j) \ ,$$

calculates its weight correction term (used to update vij later),

(19) $$\Delta_{vij} = a\delta_j x_i$$

and calculates its bias correction term (used to update vij later),

(20) $$\Delta_{v0j} = a\delta_j$$

- ***Update weights and biases*:**

Step 8: Each output unit $(Y_k, k = 1, \dots, m)$ updates

its bias and weights $(j = 0, \dots, p)$*:*

(21)
$$W_{jk}(new) = w_{jk}(old) + \Delta w_{jk} \ ,$$

Each hidden unit $(Z_j, j = 1, \dots, p)$ updates
its bias and weights $(i = 0, \dots, n)$*:*

(22)
$$v_{ij}(new) = v_{ij}(old) + \Delta_{v_{ij}}$$

Step 9: Test stopping condition met.

3.3.2 Limitations of the Multilayer Perceptron

In order to use an MLP to predict the future values of a sequence, we have to break the input series to smaller samples that overlap each other. Thus, the time steps created become variables for the MLP to approximate the function. The overlaps are simulating a window that defines the wanted output. Sutskever, Vinyals and Le (2014) argue that despite this technique may be very efficient under some circumstances there are certain critical limitations.

- **Stateless.** MLPs approximate a fixed function. The outputs that are depended on the sequence of the inputs can only be expressed by generalization into the weights values.

- **Unaware of Temporal Structure.** Since time steps are given as input variables the MLP only passively handles the temporal structure and the series of observations.

- **Messy Scaling**. For problems that require modeling multiple parallel input sequences, the number of input features increases as a factor of the size of the sliding window without any explicit separation of time steps of series.

- **Fixed Sized Inputs.** The sliding window must have a static size and I must be the same for all the inputs of the MLP.

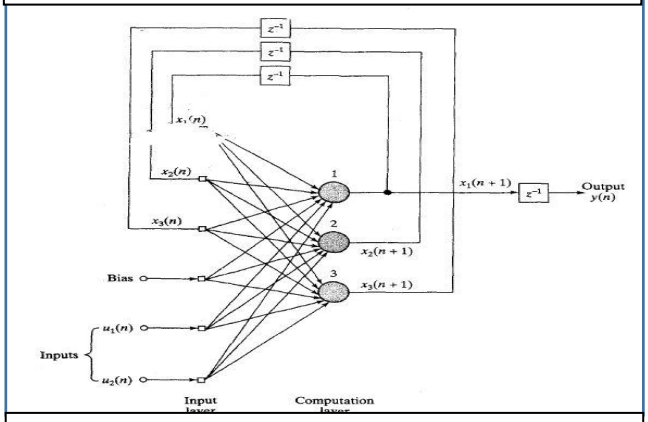
- **Fixed Sized Outputs.** The outputs must also have a static size and when this is not the case we have to manipulate them accordingly.

Despite Multilayered Perceptrons have considerable capabilities on predicting time series they are limited by the fact that the temporal dependence between observation has to be predefined in the architecture of the network. Time series prediction poses hard challenges for Neural Networks because it is required that the dimensions of variables are predefined and static.

3.4 Recurrent Neural Networks.

Recurrent Neural Networks are a type of NN that are specialized in time series data. They feature special types of nodes that have loops adding feedback and memory to the network thus they are called *memory neurons.* The memory attribute allows the RNN to approximate functions based on sequences of values rather than a static pattern (Haykin,1998).

Figure 5. The Recurrent Neural Network



Source: (Haykin, 1998)

When Back-propagation is used in very deep neural networks and in unrolled recurrent neural networks, the gradients that are calculated in order to update the weights can become unstable. They can become very large numbers called exploding gradients or very small numbers called the vanishing gradients. These large numbers, in turn, are used to update the weights in the network, making training unstable and the network unreliable. In recurrent neural network architectures, this problem has been alleviated using a new type of architecture called the Long Short-Term Memory Networks that allows deep recurrent networks to be trained (Pascanu, et al. 2013).

3.5 Long Short-Term Memory Neural Networks.

Hochreiter and Schimidhuber (1997) introduced the LSTM (long short-term memory) network as a special Recurrent Neural network type which can capture autoregressive structures of arbitrary length. Contrary to a regular RNN the LSTM doesn't need a predetermined number of "feedback loops".

The Long Short-Term Memory Neural Networks have been used to model functions for language translation applications (Sundermeyer et al., 2012), music recognition software (Eck and Schmidhuber, 2002), speech recognition programs (Graves et al., 2013) and facial motion capture (Wollmer et al., 2012).

Research results for the past three years revealed that LSTM Networks show significant Improvements compared to state of the art deep traditional multilayered perceptrons (Graves et al., 2013). Hinton et al. (2012) mention that despite the broad recent use of neural networks few academic papers are using LSTMS in their methodology.

LSTM networks use LSTM blocks as neurons similar to the system in formula (23) in contrast with feedforward neural networks. The LSTM blocks are comparable to a differentiable computer's memory chip. Every one of them consists of a group of recurrently connected memory cells and three multiplicative units. An input, an output and a forget gate that are the read, write and reset operators for the cells. The gates are the interaction mediums between the cells and the network (Graves and Schmidhuber 2005).

The LSTM block system: (23)

$$\begin{cases} f_t = g\left(W_f[x_t, h_{t-1}] + b_f\right) \\ i_t = g(W_i[x_t, h_{t-1}] + b_i) \\ o_t = g(W_0[x_t, h_{t-1}] + b_i) \\ c_t = f_t \circ C_{t-1} + i_t \circ \tanh(W_c[x_t, h_{t-1}] + b_c) \\ h_t = o_t \circ tahn(c_t) \end{cases}$$

Where: $g(\cdot)$ is the sigmoid function,

$tanh$ is the hyperbolic tangent,

x_t is the input vector,

h_t is the output vector,

c_t is a cell state vector,

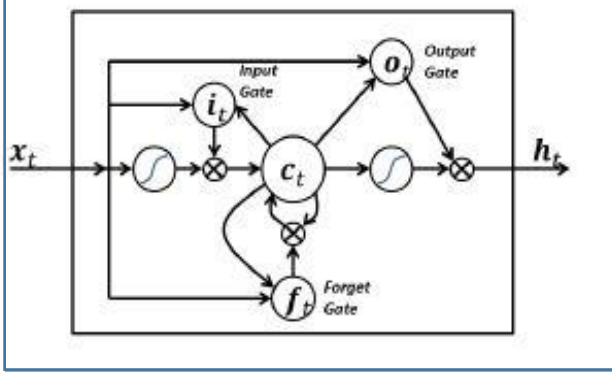
W are weights,

b are biases,

f_t, i_t, o_t are the gates of the block.

Formula (23) describes the LSTM block with its recurrent attributes.

Figure 6. The LSTM Neural Network Architecture



Source: (Hochreiter, 1997)

For example, h_{t-1} is the output vector for $t-1$ and is included in the calculation of h_t. The f_t gate of the LSTM block is the forget gate which controls the information that is to be forgotten by the cell state, c_t. The sigmoid function effectively wraps the linear expression and maps it between values 0 and 1. Whenever f_t is set to 0 by the optimizer the value is "forgotten in the calculation. The latest information stored into the cell state is controlled by gate i_t and the output comes from gate o_t. Then the cell state and the output gate are merged in output vector h_t.

According to Horcheiter and Shmidhuber (1997) the crucial part of comprehending LSTM Neural Networks is to elaborate in depth on the way the activation function regulates the memory of the nodes:

The weights of the memory cell hold parameters for the input, the output and also an internal state that depends on the input time steps.

The input weights are weighing input for the current time step while the output weights are weighing the output from the last time step.

The internal state is weighing the output for the current time step. The gates are the portals to the memory cells. They also operate As functions that have weights that additionally control the information that is driven into the cells. There are three types of gates:

- A forget gate that decides the information to be deleted from the cell.

- An input gate that chooses which inputs to use to update the memory state.

- An output gate that formulates the output based on the input and the memory of the cell.

The forget and the input gates are updating the internal state while the output gate makes the terminal decision on the output of the cell.

The gates in cooperation with the data flow are what is keeping the weights stable (neither exploding or vanishing) and internal architecture of the memory cells keeps the error flow constant.

This is basically the reason it can bridge very long-time lags. The input and output gate units learn to regulate the error flow in the memory cells. The input gate protects the error flow from deviation caused by irrelevant inputs the same ways the output gate protects the other units from deviation caused by currently irrelevant memory values (Hochreiter and Schmidhuber, 1997). The graphical representation of an LSTM complex. Contrary to the

Multilayer Perceptron the lines weights and gates are scattered in a way that is visually inconvenient to understand.

Figure 6 is a clean example found in the bibliography. The main advantages of the LSTM can be summarized as:

- They overcome the RNN limitations of vanishing and exploding gradients.

- Use memory in a way that handles the issue of long-term dependency of input sequences.

- Allow processing input and output sequences each time step at a time, allowing different length inputs and outputs.

3.5.1 Limitations of the LSTM Networks.

LSTM is a marvelous technique to capture the autoregressive attributes of time series. Nevertheless Gers, Eck and Schmidhuber (2002) state that the majority of the research concerning time series testing doesn't need to involve the complexity of neural networks methodology because the memory that is required to explain the future values can be included in only a few past observations requiring a small-time window. Gers, et al. (2002) support that LSTMs are very memory intensive requiring the nodes to store observations over a very long number of input time steps. Thus, in case that the time window is small enough a simple autoregressive model or a feedforward Multi-Layer Perceptron may perform better since they can input the whole data at once.

3.6 Comparison Models.

In this section, we will present the algebraic expression of some comparative forecasting models we will going to use. This is for the purpose of comparing the results of an investment strategy based on predictions of a novel methodology like LSTM is, to the static feed forward Multilayered Perceptron and to predicting methods that have been well adopted by investors for a long time now.

3.6.1 A Multilayered Perceptron Model using time window

Our first comparison model will be a Multilayered perceptron operating as seen in sections 3.3 and 3.3.1. Dunis et al (2012) demonstrate how the Multilayered Perceptron can handle time series problems using recent time steps (lags) as inputs with the most recent instance of the data as output to predict future values.

3.6.2 Moving Average Crossover model.

The Moving Average Crossover model is simple. We create two moving average series with different lengths. When the smaller moving average intercepts the big moving average from below in the graph we take a "long" position and vice versa (Vasilakis et al, 2012). The moving average periods choice depends on the judgment of the trader.

For our purpose, we use the movement of the index itself and its seven-day moving average so it's a 1,7 MAC strategy.

The moving average model is defined as:

(24)

$$M_t = \frac{(Y_t + Y_{t-1} + Y_{t-2} + \cdots + Y_{t-n+1})}{n}$$

Where: M_t Is the moving average at time t,

n the number of terms in the moving average,

Y_t the actual rate of return at period t.

3.6.3. The ARMA model

It is considered important to use a time series model in comparison with the LSTM model for the sole reason that financial time series data is traditionally analyzed using Econometric Time series Models.

The ARMA model is the combination of an autoregressive model with p, AR(p) lags with a moving average model with q, MA(q) lags. It assumes that the value of a time series depends on previous values (autoregressive) and on previous residual values (moving average) (Brooks, 2008).

The ARMA model has the form:

(25)

$$Y_t = \varphi_0 + \varphi_1 Y_{t-1} + \varphi_2 Y_{t_{-2}} + \cdots + \varphi_p Y_{t-p} + \varepsilon_t - w_t \varepsilon_{t-1} - w_2 \varepsilon_{t-2} - \ldots - w_q \varepsilon_{t-q}$$

Where:

Y_t is the dependent variable at time t,

Y_{t-1}, Y_{t-2} and Y_{t-p} the lagged dependent variable,

φ_0, φ_1, φ_2 and φ_p the regression coefficients,

ε_{t-1}, ε_{t-2}, and ε_{t-p} the previous values of the residual and

w_1, w_2 and w_q the weights.

3.7 Methodology Application and Justification

In order to investigate our hypotheses in practice, we will build an LSTM Artificial Neural Network structured by the specifications of the research questions to render a forecasting algorithm.

The output of the Neural Network will be a classification function that can make a forecast by classifying future direction of the market based on our data inputs. Another program in a fin-tech network can take this forecast as input to help decision making (Balch and Romero 2014).

For this dissertation, we are going to use the Python programming language (Zelle, 2004) to construct the neural network. Python programming language is developing at the moment as the standard framework for data analytics and machine learning for finance (Hilpisch, 2014).

The platform that we will be using is the Anaconda data science suite which is a condensed open source environment including over 100 of the most popular Python packages for data science. (Boschetti and Massaron, 2015)

Our calculation framework will be the Tensor Flow Machine Learning Library module – an open source data-flow based programming model developed by Google from real life experience by over one hundred machine learning products and services (Abadi et al, 2016). According to Goldsborough (2016) Tensor flow can perform fast automatic gradient computations and adds a lot of new features compared to the previous machine learning toolkits as Theano and Torch. He notes that since little investigation has been done in literature to evaluate Tensor flow qualities it's a good study topic for the academic community in the future. He states that Google's platform will be a great benefit for the scientific community as a whole opening a new door to faster larger-scale artificial intelligence.

In part 3.7.1 the experiments that consist the methodology are presented. Parts 3.8 through 3.10.2 present the technical specifications and hyperparameters of the LSTM model and the comparison models.

3.7.1 Experiments.

The methodology is structured by five experiments,

i) Classification with a sophisticated LSTM network model to predict one day ahead direction of the market for the period given.

ii) A Moving Average Crossover technical indicator to compare predicting performance with the LSTM model.

iii) A forecast of the time series using an ARMA model to compare predicting performance with the LSTM model.

iv) A Multilayered Perceptron Model with a time window to compare predicting performance with the LSTM model.

v) A trading application will be presented based on the predictions of the models.

Financial forecasting is a very complicated and multivariate area. Consequently, the widely used MAC strategy presented in the second experiment of the methodology is not expected to have impressive result considering the EMH paradox and the copious amounts of traders using it. The same applies to the simple ARMA model presented in the third experiment that its linear nature is very simplistic to approximate the functional forms of today's financial markets movement.

In the first and the fourth experiments, we will test our Neural Network models. This is expected to be interesting due to its generalization capabilities and their efficiency with non-linear functional forms.

The fifth experiment will involve a backtesting of the produced models. The strategy will be simple, and it will the same for all methodologies. The orders will be theoretically given before market close every day and there will be to go or stay long if the

predicted returns are above zero and to go or stay short if the predicted returns are below zero. For the neural network models, the 33% of the data will be used for testing, thus not included in the forecasting dataset.

The trading performance will be evaluated simply by the percentage of profit or losses before and after transaction costs and leverage fees. The platform for the evaluation of all algorithms will be Qstrader backtesting and live trading platform for python (Quantstart, 2017).

3.8 Neural Network Models Architectures.

To Decide on the architecture of our network we have to test for the optimal "*hyperparameters*". That is the parameters that don't depend on the data and our model doesn't calculate by itself. For ANN that is the number of layers, the number of neurons on each layer, the number of epochs and the number of dropout neurons in the regularization process. in order to monitor the Artificial Neural Network performance for each set of hyperparameters and decide our architecture, we will test run different combinations and measure the performances based on Twomey's and Smith's (1995) paper suggestion of the Mean Absolute Error as presented in formula (26).

(26)

$$MAE = \frac{1}{n}\sum_{j-1}^{n} |y_j - \hat{y}_j|$$

Where: y_j is the prediction

$\hat{Y}_j$ is the true value.

(Twomey and Smith, 1995)

3.8.1 Multilayer Perceptron with a time window

The Multilayered Perceptron can handle the time series problem using a number of recent time steps as variables. In python, we can formulate our dataset using the *look_back* argument in the *create_dataset()* function. The network will consist of five layers with three hidden layers having twenty-four neurons in the first layer, twelve neurons in the second layer and eight neurons in the third layer.

Table 1. Hyperparameters for MLP Network.

Hyperparameter	Value
Learning algorithm	Adaptive Moment Estimation (ADAM)
Loss Function	Mean Average Error
Learning rate	0.001
Momentum	0.003
Batch size	2
Epochs	200
Initialization of weights	N (0.1)
Input Neurons	1
Hidden Layers	3
Hidden Neurons	24, 12, 8
Output Neurons	1

3.8.2 The Long-Short Term Memory Network

Since we will be using a sophisticated LSTM model for evaluating our hypothesis we will use a wider topology within each layer. Its architecture will be consisted of three hidden layers of neurons.

The first hidden layer will include 4 neurons, the second 50 and the third 100. The last two layers are going to be optimized by a dropout regularization layer which will drop half of the neurons in each epoch. The output layer is going to be wrapped by a SoftMax function as in equation (27).

(27)

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{k} e^{z_k}} \quad \text{for} \quad j = 1, ...,\mathrm{K}$$

3.8.3 Dropout Regularization

Artificial Neural Networks can be big and complex so that they can produce algorithms that many times overfit the training data. This can render them useless into forecasting future data that is not perfectly similar to the one we used for training. Thus, we have to be careful to tradeoff overfit and underfit effects in order for our model to be functional (Dunis et al, 2002).

Srivastava, et al. (2014) propose a technique in order to prevent overfitting which they call "*Dropout Regularization*". Their technique is to randomly select nodes which are ignored during training, so they don't participate in the activation of the neurons

on the forward pass and their weight updates don't apply to the phase of backpropagation. Thus, the NN has to find an "intelligent way to a more generalized function to describe the data and can predict better future instances of the same data-set.

3.8.4 The ADAM Learning Algorithm

The learning algorithm of choice for our artificial neural networks is the adaptive moment estimation (ADAM) method.

The ADAM algorithm is introduced by Kingma and Ba (2015). They claim that the Adaptive Moments Estimator (ADAM) algorithm is very efficient due to the fact that it uses only first-order gradients. This ensures that the algorithm uses minimal memory while achieving optimal performance.

They state that the algorithm combines the advantages of two widely used algorithms namely the Adaptive Gradient algorithm (ADAGRAD) (Duchi et al., 2011) which operates optimally with sparse gradients and the Root Mean Square Propagation algorithm (RMSProp) (Tieleman & Hinton, 2012), which is better for online and non-stationary settings.

Further ADAM method benefits include that the magnitudes of the parameters updates are uncorrelated to the scale of the gradient, its step sizes are approximately bounded by the step size hyperparameter, it does not require a stationary objective, it works with sparse gradients and it naturally performs a form of step size annealing (Kingma and Ba, 2015).

Table 2. Hyperparameters for LSTM Network.

Hyperparameter	Value
Learning algorithm	Adaptive Moment Estimation (ADAM)
Loss Function	Mean Average Error
Learning rate	0.001
Momentum	0.003
Batch size	1
Epochs	200
Initialization of weights	N (0.1)
Input Neurons	1
Hidden Layers	3
Hidden Neurons	4, 50, 100
Output Neurons	1

3.9. Amazon Elastic Computing Cloud

For our calculations, we will use Amazons EC2 service to create a G2 instance type cloud server ideal for machine learning training.

Varia and Mathew (2014) present an overview of the Amazon Web Services. AWS provide the capability to create scalable on-demand servers online for our intensive applications for a small fee. We can use and pay only the resources we need as long as we need them. The g2.8large instance involves four NVIDIA GRID GPU's each with 1,536 CUDA cores and 4GB video RAM. It also includes 32vCPU's, 60 GiB of memory and 240 GB of SSD storage. It is a very powerful computing platform that allows us to speed up training and tune our hyperparameters efficiently with a cost of only 0.60 $ per hour of use.

3.10 Comparison models

In this part of the Thesis, the empirical methodology of the comparative models is presented. As mentioned before addressing this comparison is crucial since financial market forecasting traditionally involves decision-making processes similar to the following.

3.10.1 ARMA model

We used the E-views econometric software to formulate our ARMA model. According to the auto-ARIMA function process we ended up with an ARMA (3,4) model. This also is the number of time steps that the LSTM NN will be operating. The coefficients of the autoregressive and the moving average terms have statistical significance 99%

Table 3. The ARMA model calculation.

Dependent Variable: RETURNS
Method: ARMA Maximum Likelihood (OPG - BHHH)
Date: 12/09/17 Time: 02:46
Sample: 1/22/2001 10/20/2017
Included observations: 4151
Convergence achieved after 53 iterations
Coefficient covariance computed using outer product of gradients

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.000309	0.000346	-0.891376	0.3728
AR(1)	0.457292	0.038072	12.01131	0.0000
AR(2)	0.238894	0.051545	4.634639	0.0000
AR(3)	-0.888473	0.037436	-23.73327	0.0000
MA(1)	-0.387996	0.040558	-9.566404	0.0000
MA(2)	-0.305940	0.045270	-6.758124	0.0000
MA(3)	0.893866	0.027999	31.92536	0.0000
MA(4)	0.039722	0.013981	2.841092	0.0045
SIGMASQ	0.000454	5.24E-06	86.51657	0.0000

R-squared	0.016306	Mean dependent var	-0.000309
Adjusted R-squared	0.014406	S.D. dependent var	0.021479
S.E. of regression	0.021324	Akaike info criterion	-4.855746
Sum squared resid	1.883372	Schwarz criterion	-4.842019
Log likelihood	10087.10	Hannan-Quinn criter.	-4.850890
F-statistic	8.582267	Durbin-Watson stat	2.000253
Prob(F-statistic)	0.000000		

Inverted AR Roots	.68-.72i	.68+.72i	-.90	
Inverted MA Roots	.68-.72i	.68+.72i	-.04	-.93

source: E-views

For our data, the specific ARMA model is formed as:

(28) $$Y_t = -3.09x10^{-4} + 0.457Y_{t-1} + 0.238Y_{t-2} - 0.888Y_{t-3} - 0.387\varepsilon_{t-1} - 0.305\varepsilon_{t-2} + 0.893\varepsilon_{t-3} + 0.039\varepsilon_{t-4}$$

3.10.2. The Moving Average Crossover model.

We will use Qstrader back-testing and live trading engine in python to run and evaluate our MAC model (Quantstart, 2017).

As mentioned in part 3.7.2 our moving average terms will be the index itself and the seven days moving average. This practically means that the strategy will place a buy or stay long order when the line representing the daily closing value of the index crosses its seven-day moving average from below or the opposite.

Chapter 4. Data analysis and Findings

In the fourth part of this Thesis, we present our data and the findings of our experiments. In part 4.1.1 we present a brief description of the preferred attributes for a financial dataset. In section 4.1.2 the characteristics of the dataset along with its linear graph, its histogram and statistics is discussed. In parts 4.2 through 4.2.1 we evaluate the prediction accuracy of the models and in part 4.2.2 we discuss the performance of our models through a back-testing application.

4.1 Data

4.1.1 Optimal attributes of financial datasets

Financial datasets are widely available today mainly because of the outspread of online sources. There are many companies that sell or distribute for free OHLC type data for every kind of financial instruments. Nevertheless, researchers should be aware and evaluate the source of the dataset they use as much as the qualitative characteristics of the dataset itself.

Balch and Romero (2014) state a number of characteristics a financial dataset should have in order to be able to be used for forecasting purposes.

- You should have access both to historical and future data in order to be able to evaluate by backtesting and forward testing your algorithm.

- 2) The data must be "survivor bias-free" meaning that it should exclude assets that don't exist anymore (delisted stocks, bankrupted companies).

- 3) Integration with the trading platform or backtesting algorithm should be seamless. There should be access to low-latency real-time feed of the data in order to implement the algorithm in real-time trading.

4.1.2 The FTSE/ATHEX Large Cap Data

The FTSE/ATHEX Large Cap index allows us to be diversified while being exposed to the top 20 shares traded on the Greek Stock Exchange. It was developed in 1997 by the partnership of Athens Stock Exchange with FTSE International and is the established reference to the Greek stock market performance (Dunis, et al. 2011).

As of 2 July 2017 the stocks comprising this index are: Viohalco, Coca-Cola HBC AG, Gr. Sarantis S.A, Titan Cement, Ellaktor, Alpha Bank, National Bank of Greece, Eurobank Ergasias, GEK Terna, Mytilineos Holdings, Lamda Development S.A., Jumbo S.A., Folli Follie, Hellenic Petroleum, OPAP, Piraeus Bank, Motor Oil Hellas, Public Power Corporation, Piraeus Port Authority (OLP), Grivalia Properties R.E.I.C., Fourlis S.A., Hellenic Exchanges Group, Aegean Airlines, ADMIE, OTE, Metka (Athens Stock Exchange, 2017).

For practical purposes and ease of handling we can use our Artificial Neural Network algorithm to trade the electronically traded stock market index futures contract FTSE/ATHEX Large Cap tracking the FTSE/ATHEX Large Cap index.

Futures are contracts for a specific transaction involving an underlying asset at a fixed price in a fixed date that can be traded allowing us to follow the movement of the value of the underlying asset in the market (Hull, 2008). As the FTSE/ATHEX Large Cap index futures contract is traded in index points its price is the points multiplied by 2 euros (Athens Stock Exchange, 2017). Thus, a contract trading at 1975 points is valued 3950 euros. Therefore, the FTSE/ATHEX Large Cap Futures makes it easier to trade the index for a small cost with less transactions.

According to Fassas (2010), FTSE/ATHEX-20 index futures diverge significantly from the spot price of the index presenting opportunities for arbitrage. This might create a substantial tracking error that would affect a passive strategy. Nevertheless, the tracking error is not affecting our active strategy performance since what we are interested in is the movement of the selected instrument based on our prediction.

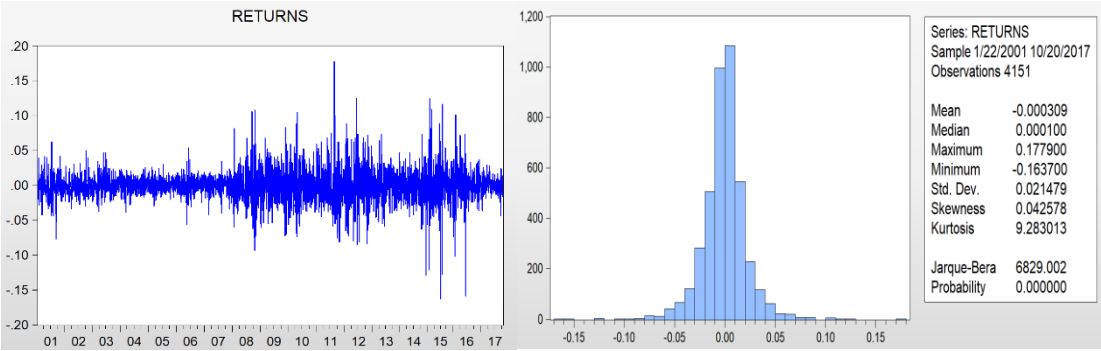
The data that will be used for the input values of the model was downloaded from investing.com. Investing.com offers essential Historic financial and economic data alongside with utilities to do technical and fundamental research about our asset of interest (investing, 2017).

Python modules offer the possibility to directly stream data for forecasting purposes after we train our Network. The frequency of the data in our research will be daily and the interval will be 17 years approximately from 21 January 2001 to 20 October 2017 (table 4.). The input data unit will be in ECB Euros.

Graph 1: FTSE/ATHEX Large Cap movement

Source: investing.com.

Graph 2: Data-set distribution and statistics.



Source: E-Views.

We will use the logarithmic returns of the data. Logarithmic financial returns are assumed to have its characteristic roots in the unit circle, therefore, being stationary. Is visually evident that the daily FTSE/ATHEX Large Cap dataset distribution is not normal (Graph. 2). It is leptokurtic according to the 3rd stylized fact of financial data in section 2.4 and has a positive skewness.

As our input variables are time steps of a single returns series with a constant daily frequency it does not need any further preprocessing or rescaling to be used by our machine learning models.

Table 4. The datasets.

Name of period	Trading Days	Beginning	End
Total dataset	4152	22 January 2001	20 October 2017
Training dataset	2782	22 January 2001	13 March 2012
Test dataset	685	13 March 2012	07 November 2014
Out of sample dataset	712	07 November 2014	20 October 2017

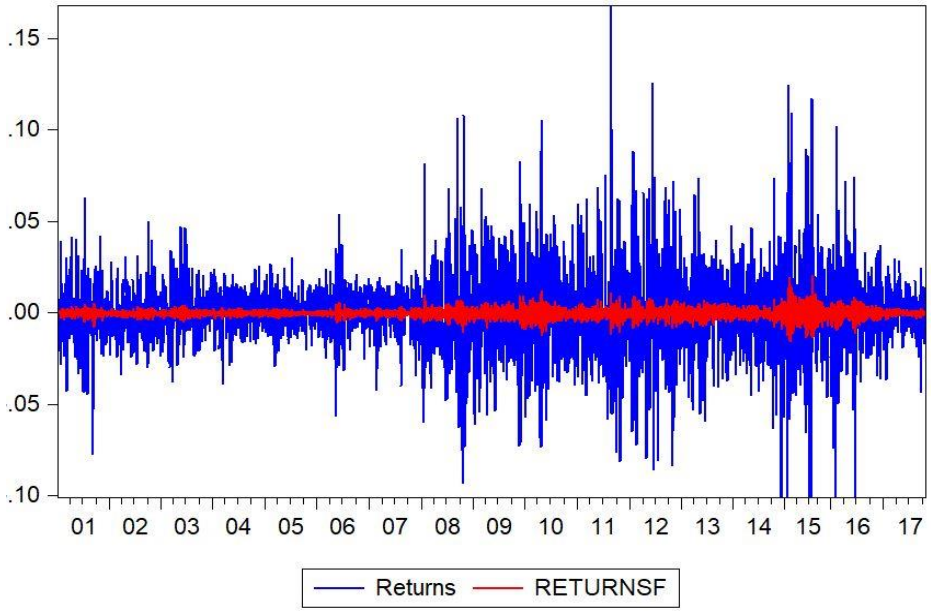
Table 5. Explanatory variables for ANN.

Number	variable	lag
1	FTSE/ATHEX Large Cap returns	1
2	FTSE/ATHEX Large Cap returns	2
3	FTSE/ATHEX Large Cap returns	3

4.2 Results

Graph 3 shows the predictions of the ARMA model (red) compared to our actual time series. We can visually observe that the model follows the direction of the actual data but is not capturing the magnitude of the movement. We can also observe that it captures the high volatility clusters of the series occurring in the period after 2008. The performance is not bad considering that it is a linear model that is not adaptive to the functional form of the data.

Graph 3. The ARMA model forecast results

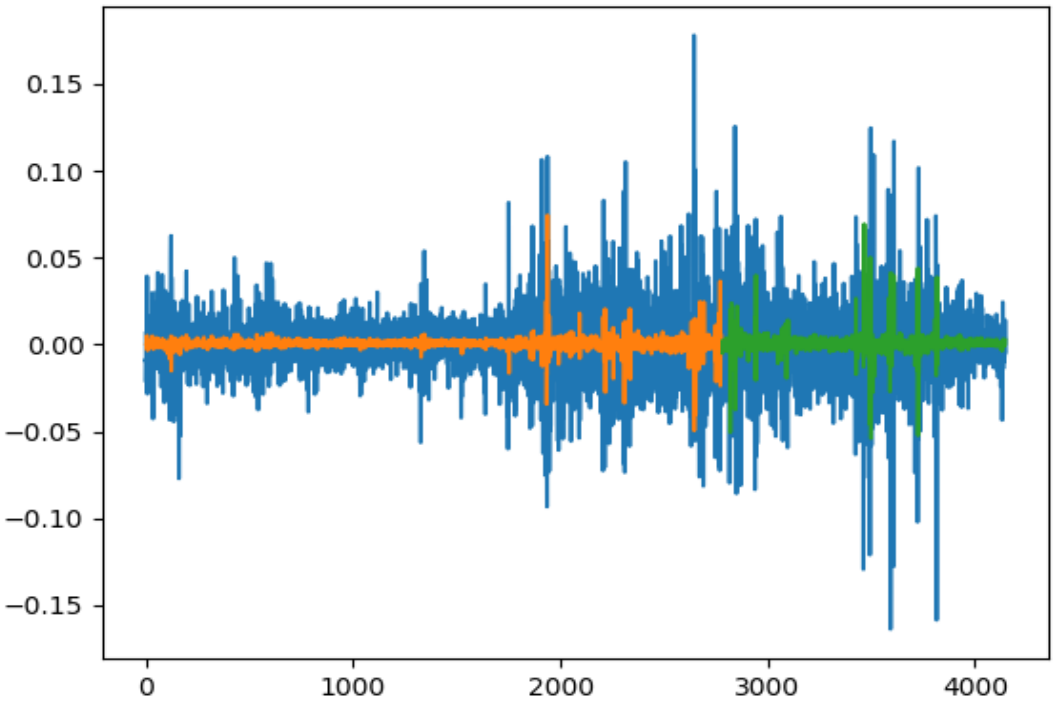


Graph 4 shows the predictions of the MLP model on the training (orange) and the testing (green) part of our sample. Comparing the MLP graph to the ARMA model we can see that the neural network captures more efficiently the volatility across the dataset.

Especially in the last clusters of the testing sample, the performance is impressive. As mentioned in part 3.3.2 the Multilayered perceptron has several limitations when it comes to time series prediction mainly due to the static nature of its dependencies.

Nevertheless, as it is mentioned in part 3.5.1 Multilayered perceptron's can perform equal or better in situations that small time windows are used, and all the data involved is fed to the input layer at once. In situations like that the simplicity of the MLP neuron compared to the LSTM's can save us computational time and memory.

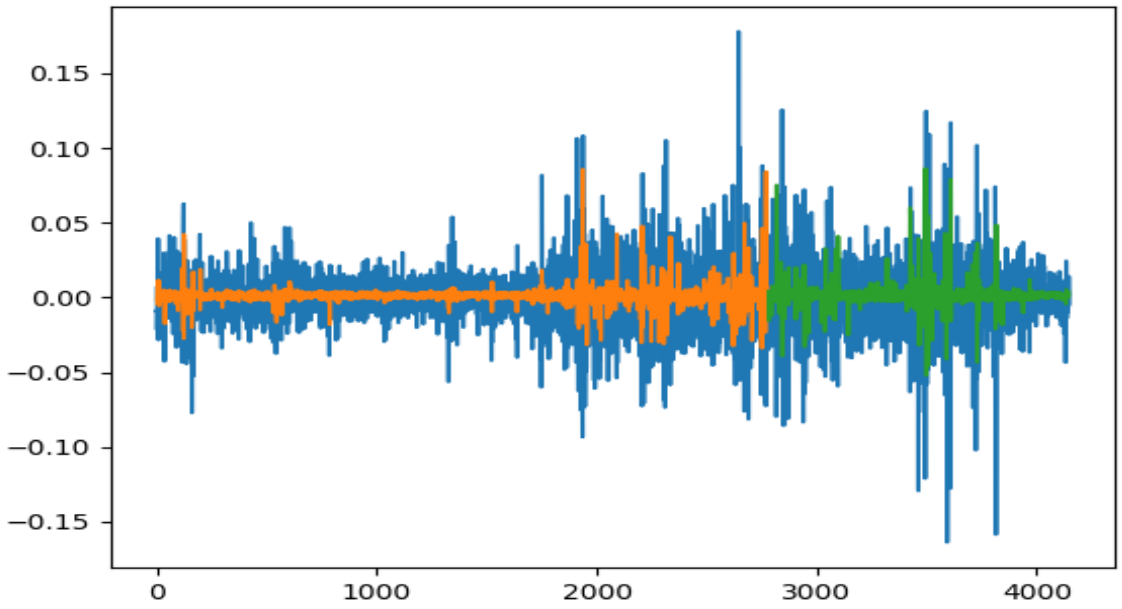
Graph 4. Multilayered Perceptron forecast results



Graph 5 shows the predictions of the Long Short-Term Memory network. Like the Multilayered Perceptron above the model profoundly detects the volatility clusters on the dataset making it a very promising forecasting tool for volatility-based trading strategies.

We can also observe how the LSTM captures more effectively the magnitude of the volatility clusters compared to the MLP model. As mentioned in part 3.5 the LSTM handles better the dependencies between time steps due to the way its nodes treat and store memory over time thus it can approximate the functional form in a dynamic way compared to the static Multilayered perceptron,

Graph 5 Long-Short Term Memory Network forecast results



4.2.1. Forecast evaluation

As suggested by Twomey's and Smith's (1995) paper we use the Mean Absolute Error (MAE) as presented in formula (26) to evaluate the predictive accuracy of our models. Along with that, we calculate The Mean Squared Error (MSE) and the Root Mean Squared Error (RMSE). All These measures interpret the deviation of the forecasted value to the actual value as a positive number.

Table 6. Out of sample statistical results

Evaluation Formula	MLP	ARMA	LSTM
MAE	0,013756	0,014658	0,010845
MSE	0.016456	0,026382	0,011474
RMSE	0.015537	0.025306	0.012678

Contrary to Mean Absolute Error that weights equally all deviations to give the absolute mean, the Mean Squared Error and Root Mean Squared error are squaring the errors. As a result, they weight heavier the highly erroneous observations (Twomey and Smith, 1995). This makes the latter a good evaluation tool for models that instantaneous volatility matters the most. According to the above and the results on section 4.2 we can observe the values of the accuracy measures to differ more in the case of the ARMA model compared to the neural networks models due to the ability of the latter to capture more efficiently the volatility clusters on the dataset.

4.2.2. Strategy Back Testing Evaluation

Once we develop a model that can support a trading strategy it is wise to test it before we invest funds. A realistic approach would be to "forward test" the model using simulated trading for some time in the future.

Forward testing can be very time consuming while conditions may change, and the market might get efficient towards our model in the meanwhile. A faster way is to test our strategy and predictive model over past data. This is called "backtesting".

According to Balch and Romero (2014), the main components that are required to conduct a backtest are"

- Historical data over which the strategy will be evaluated
- 2) An algorithmic trading strategy
- 3) An algorithmic trading platform that supports realistic market dynamics such as commissions and market impact cost.

- 4) Analysis of the returns of the algorithm and comparison to other benchmark alpha generating methodologies. These four components are enough to evaluate how a strategy would perform during a historical timeframe and analyze its returns sources and its risk factors.

4.2.2.1 Backtesting Biases

Balch and Romero (2014) also pinpoint the hazards that can arise from using backtesting evaluation methods and summarize the main fallacies an analyst can fall into. They recognize the three main misconceptions of back-trading as being

- Unconsciously picking into our data and creating an algorithm that describes perfectly the functional form. This would overfit the dataset creating highly accurate predictions, but it would only create abnormal returns in the future by pure luck due to its inability to generalize the functional form of new data.

- 2) The data mining fallacy occurs when we develop our algorithm by testing a mass of different methodologies on the same data. Inevitably we will find the best fitting algorithm which will be having trouble to generalize and will fit any forward testing situation only by luck.

- 3) Changes in the market dynamics is another consideration that derives directly from the Efficient Market Hypothesis. An algorithm developed on data from the late 90s would always choose to go long on technology stocks which might perform averagely in today's circumstances.

Another very important bias that can affect algorithmic trading results is Survivor Bias. Elton et al. (1996) highlight the effect of survivor bias in the estimation of mutual fund performance. They describe how survivor bias phenomenon can occur when we develop an algorithm exclusively on past data of companies that survive today ignoring all the failures. This can significantly inflate the results of our back-testing evaluation upwards since in fact future information is being used to predict past performance. We can avoid back-testing fallacies by using cross-validating data sets on our algorithms. This is testing our algorithm on multiple data sets which represent various time periods and different economic conditions.

4.2.2.2 Backtesting benchmarks and results

In table 7 we present the trading performance of all our models. According to our methodology, the strategy simply operates in compliance to the most probable class proposed by the classification model. As mentioned before the strategy goes or stays long if the classifier forecasts positive returns or goes or stays short if the classifier forecasts negative returns. All algorithms are evaluated on the same exact strategy for comparison purposes.

The evaluation benchmarks are the annualized returns of the particular strategy for each algorithm compared to the annualized volatility and the maximum drawdown. The Sharpe ratio is wrapping up all the above presenting the risk-adjusted return of the algorithm (Sharpe, 1994). We also present the number of positions taken by each algorithm annually. This is important as it raises the transaction and taxation costs minimizing the realized profit.

The Sharpe Ratio is presented in formula 29.

$$SR = \frac{E(R_i) - RFR}{\sigma} \qquad (29)$$

Where: SR = Sharpe Ratio

$E(R_i)$= expected return of the instrument

RFR = return of the risk-free rate

σ = overall risk of the asset expressed as volatility

To estimate the Sharpe ratio, we use as risk-free rate the 1-year Euribor which is the average interest rate at which Eurozone banks offer to lend funds to each other.

Table 7. Trading performance results.

Evaluation Formula	BUY&HOLD	MLP	MAC	ARMA	LSTM
Sharpe ratio (excluding cost)	-0.73	0.55	0.24	0.58	0.62
Annualized Volatility (excluding cost)	40.50%	40,80%	42,10%	40,30%	41,20%
Annualized Return (excluding cost)	-30.00%	22,40%	10,00%	23,30%	25,50%
Maximum drawdown (excluding cost)	-65,00%	-30,50%	-41,00%	-40,10%	-35,50%
Position taken (annualized)	1	95	70	140	105

We can observe that for all our models the annualized volatility is clustering around 40%. That makes it easier for us to compare the results in terms of returns as the risk is very similar. The annualized loss is big for the buy and hold strategy given that our data starting point is at 2001 right after the first Greek stock market bubble burst. Following where the second Greek stock market bubble and the Global financial breakdown of 2008.

The annual returns for the Neural networks and the ARMA model are clustering around 20% with the LSTM being the most profitable reaching up to 25.50%. The Moving Average Crossover strategy scores 10% following the crossing over-under rules whereas all the other strategies re-evaluate their positions before the end of every trading day.

The maximum drawdown is significant for all algorithms and is considered unsustainable for realistic trading conditions. Considering the positions taken we can see that the Neural Networks models have taken 95 and 105 compared to the 140 of the ARMA model. This will impact transaction and taxation costs.

4.2.2.3 Trading Costs and Leverage.

As the futures contract is traded in index points its price is the points multiplied by 2 euros (Athens Stock Exchange, 2017). Thus, a contract trading at 1975 points is valued 3950 euros.

Athens Stock Exchange charges 0.70 euros of transaction commision per contract for a minimum of 251 of FTSE/ATHEX Large Cap futures contracts. In today's market values 1000.000 euros buys us approximately 253 contracts which divided by an average volume deal of 1000.000 gives as an average transaction cost for institutions and fund managers of about 1.77basis points or 0.0177% per position.

Transaction cost is a considerable parameter in frequent trading strategies thus we could use a restriction for our model that prevents a trade to happen if the returns considering transaction costs are equal or below zero.

Table 8. Trading performance results with transaction cost.

Evaluation Formula	MLP	MAC	ARMA	LSTM
Sharpe ratio (excluding cost)	0.55	0.24	0.58	0.62
Annualized Volatility (excluding cost)	40,80%	42,10%	40,30%	41,20%
Annualized Return (excluding cost)	22.40%	10,00%	23,30%	25,50%
Maximum drawdown (excluding cost)	-30,50%	-41,00%	-40,10%	-35,50%
Position taken (annualized)	95	70	140	105
Transaction costs	1,68%	1.23%	2.47%	1.85%
Annualized Returns (Including Cost)	20,72%	8.77%	20.83%	23.65%

Concluding our strategy performance presentation, we can present a method to raise the profitability of our model by using a leverage coefficient. The leverage coefficient is equal to all models, so it doesn't affect volatility. The interest paid for the leverage is assumed to be 4% per annum.

Table 9. Trading performance conclusive results.

Evaluation Formula	MLP	MAC	ARMA	LSTM
Sharpe ratio (excluding cost)	0.55	0.24	0.58	0.62
Annualized Volatility (excluding cost)	40,80%	42,10%	40,30%	41,20%
Annualized Return (excluding cost)	22.40%	10,00%	23,30%	25,50%
Maximum drawdown (excluding cost)	-30,50%	-41,00%	-40,10%	-35,50%
Position taken (annualized)	95	70	140	105
Transaction costs	1,68%	1.23%	2.47%	1.85%
Leverage	x2	x2	x2	x2
Leverage Costs	4%	4%	4%	4%
Annualized Returns (Including Cost)	20,72%	8.77%	20.83%	23.65%
Annualized Returns (Including cost & Leverage)	16.72%	4.77%	16.83%	19.65%

Chapter 5. Discussion and Conclusion

5.1 Discussion of the research queries.

- Has the Long-Short Term Memory Architecture Deep Neural Network written in python and using the Tensor-flow library the forecasting capability to challenge the weak form of Efficient Market Hypothesis using as data the daily returns of the Athens Stock Exchange Large Cap index for the period of 22 January 2001 to 20 October 2017?

The LSTM Neural network shows significant forecasting results when used as a binary classifier to predict the Greek stock market index futures contract (FTSE/ATHEX Large Cap).

There is no definite explanation to why a function approximator like that is efficient or not because the window of possibilities is huge. It might be true that our model generalized very well the attributes of the particular market. Another scenario might be that our training and evaluation dataset happen to be too coherent. We can also consider whether the technology used is too novel to have efficient counterparts in the Greek market so that the EMH paradox doesn't apply.

The results seem to be in accordance with Graves et al. (2013) and the literature presented in part 1.3 where most of the papers seem to agree that the LSTM networks produced more accurate algorithms compared to the rest of the models

- How does the Long-Short Term Memory Deep Neural Network compare to a Multilayered Perceptron using a time step window for the same dataset?

The Multilayered perceptron is not an architecture dedicated to time series prediction. Nevertheless, it can be used to approximate autoregressive models with success when the temporal dependence of observation is given by the labeling of the data and is fixed across the set.

For our experiment, the Multilayered perceptron with a time step window seems to efficiently capture patterns on the training part of the dataset that get to fit the testing part quite effectively. Nevertheless, as also observed in literature presented in parts 1.3 and 2.7 the MLP didn't overperform the LSTM neural network.

In accordance to Gers, et al. (2002) It's important to mention that we used a relatively small number of lags on both networks. This helped significantly the Multilayered Perceptron to compete with the Long Short Memory network and give comparably efficient results.

- How does a trading strategy following the suggested neural network forecasts perform compared to following a moving average crossover forecast?

Our results confirm that the Neural Networks model outperforms the MAC strategy. This was expected given that the Greek stock market conditions are not considered ideal for a trend-based strategy due to its long-term seesaw movement and its sudden volatility outbursts.

Tapa A., Yean S.C and Ahmad S. N. (2016) state that the Moving Average Crossover has certain limitations that affect its performance, especially when tested for large timeframes like the one on our dataset. In periods that market is not trending the moving average is moving sideways creating a lot of false signals that severe its performance. Also, volatility clusters can create noise that affects moving average causing it to create false signals. In general, moving average crossover is a good choice for markets that follow trends and have a fixed volatility pattern.

According to our results and to the large part of the literature the LSTM neural network is performing better than traditional technical analysis indicators. This is contributed mainly to their capability to capture non-linearities and to their versatility to approximate any functional form.

- How does a conventional time series ARMA model forecasting performance compare to the suggested neural network models?

As said previously Artificial Neural Networks are very efficient in capturing nonlinear patterns and temporal structures. In accordance with Ghiassi et al. (2005) and Dunis et al. (2012) the linear ARMA is difficult to reach the same levels of accuracy due to its static nature and its predetermined hyperparameters. Despite the lower accuracy levels our back-testing results were

positive for the ARMA model and were comparable to the Neural Networks models. As with the neural network models we can attribute this partially to the limitations of the back-testing procedure for a model that is fitted using past values. Another parameter that stands always in the realm of stochastic models is mere luck.

5.2 Concluding Remarks

In this Thesis, we tested Efficient Market Hypothesis using an LSTM Neural Network. We also compared its predictive accuracy with a traditional econometric ARMA approach, the Moving Average Cross-Over technical indicator and a Multilayered Perceptron using a time window with previous time steps as input features.

It seems that the LSTM network produced the model with the lowest mean average error of all the comparison methodologies reaching down to 0,010845. Additionally, we use the LSTM model along with the comparison ones to test their returns using a trading strategy according to the signal produced by the Neural Network approximated algorithms and the rest of our comparison models.

The LSTM seems to outperform the ARMA, the MAC and the MLP model in returns both before and after transaction costs.

Comparing our experiment results we can observe that both the LSTM and our comparison models challenged the Efficient Market Hypothesis for the Greek FTSE/ATHEX Large Cap Index Future.

Since the EMH is purely an empirical benchmark which depends deeply in the methodology used, it can be assumed that our model's performance is partially attributed to the fact that the

Greek market might not be exposed to this technology as much as other larger markets. Another assumption that can be made is that the specific dataset fitted very well in the out of sample data making the specific asset look inefficient. We also have to take into consideration that Neural Networks are stochastic function approximators. This practically means that different runs of the same algorithm can produce different results.

Looking back at the dataset we can also mention that since it is a free publicly distributed instance it might include "biases" or mistakes that boost the back-testing results. Also, our back-testing results cannot take into consideration the collocation effect on the performance of our algorithm. Although our tests did not include high-frequency trading scenarios in the real world the impact of co-location along with the market impact of large HFT firms would possibly greatly affect our returns even if our algorithm forecasts the direction of the market.

According to our experience to develop a practical application for trading a lot more considerations have to be taken into account. We suggest that our methodology can be further researched by thoroughly investigating the alternative hyperparameters of the LSTM neural network and testing different optimization techniques. Also, a research on explanatory variables of different nature than past returns would be interesting. These could be qualitative variables like sentiment and fundamentals data. Another suggestion would be to compare the LSTM to a diverse set of comparison models suggestively other Neural Network Types or Machine learning techniques like decision trees and support vector machines.

Concluding we shall make some remarks regarding the application of the Efficient Market Hypothesis in practice and the use of artificial intelligence to analyze and exploit financial markets.

According to the theory, the academic literature and the techniques presented in this thesis it is apparent that Efficient Market Hypothesis stands in dependence of the maturity of the market analyzed and exogenous factors like the regional and global socioeconomic status. The literature shows that the results for machine learning and artificial intelligence techniques are optimal for underdeveloped markets involving significantly lower liquidity, trading volume and a vague regulatory frame.

Another consideration is the impact of behavioral parameters in trading. This is a broad topic that includes the market sensitivity to public sentiment, the institutional investors involved, the amount of governmental intervention along with other variables.

As with all forecasting methods legal or not, public access to technology and data plays a crucial role in the efficiency of the market. Markets, where retail investors are less familiarized with novel statistical analysis concepts, are more vulnerable to exploitation and have slower "reflexes". Thus, the efficient market paradox mechanism mentioned in section 2.1 delays leaving substantial time window for reaction.

In cases of a weak regulatory frame, strong form inefficiencies may occur. This could result in preventing forecasting algorithms to create efficient signals. The same can stand for markets that are heavily affected by high-frequency trading with collocated servers.

We focus our research on the Athens Stock Exchange by training our forecasting models with past values of the ATHEX 20 index. The Greek financial market has been in turmoil the past 20 years facing regulatory and technical issues.

Focusing on the long-term patterns of our data we can see that it consists mainly of bubbles formulated by public sentiment factors and fueled by the inefficient inflationary policies of the past years. In 2009 the global financial crisis put an end to this sequence, dropped the market in unpreceded low levels and at the moment it is moving sideways driven mainly by the sentiment about the viability of the Greek banking sector. (Kouretas and Vlamis, 2010)

We can ascertain from the above that the Greek market has several characteristics that might render it inefficient according to the three forms of EMH. Consequently, we can assume that it is not surprising that at least as for the back-testing results it was easy for our forecasting models to perform as good.

Wordcount:15050

References

[1]. Abadi, M. and Barham, P. and Chen, J. and Chen, Z. and Davis, A. and Dean, J. and Devin, M. and Ghemawat, S. and Irving, G. and Isard, M. and Kudlur, M. (2016) *Tensor Flow: A system for large-scale machine learning.* In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA.

[2]. Ali, S.F. and Abdelnabi, S.M. and Iqbal, H.H. and Weni, H. and Omer T.A (2016) Long Run Relationship between Macroeconomic Indicators and Stock Prices: The Case of South Africa. Journal of Internet, Banking and Commerce, 21(52), 1-16.

[3]. Armano, G. and Marchesi, M and Murru, A. (2005) A hybrid genetic-neural architecture for stock indexes forecasting. *Information Sciences*, 170(1),3-33.

[4]. Athens stock Exchange. (2017) Futures FTSE/ATHEX Large Cap.[online].Available:http://www.helex.gr/documents/10180/2216617/Product+Sheet+Future+FTSE+ATHEX+Large+Cap+_EN.pdf/76d6c234-53a1-40fc-b37b-0291ab410e16 [Accessed 18 October 2017]

[5]. Atsalakis, G. S., & Valavanis, K. P. (2009). Surveying stock market forecasting techniques–Part II: Soft computing methods. *Expert Systems with Applications, 36*(3), 5932–5941.

[6]. Bachelier, L. (1900) Theorie de la Speculation. *Doctoral Dissertation in Mathematics, University of Paris*

[7]. Buffet, W. (1976) Benjamin Graham (1894-1976). *Financial Analysts Journal, 32(6)*.

[8]. Bahrammirzaee, A. (2010) A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems. *Neural Comput & Applic*, 19(2), 1165–1195.

[9]. Balch, T. and Romero, P. J (2014) *What Hedge Funds Really Do: An Introduction to Portfolio Management*, U.S.A, Business Expert Press.

[10]. Basu, S. and Raj, M. and Tcalian, H. (2008) A comprehensive Study of Behavioral Finance. *Journal of Financial Service Professionals,* 62(4), 51-62

[11]. Boschetti, A. and Massaron, L. (2015) *Python data science essentials*, Birmingham, Packt Publishing Ltd.

[12]. Bishop, M. C. (1995) *Neural Networks for Pattern Recognition*. Cambridge, UK, Oxford University Press.

[13]. Boyd, M. and Kaastra, I. (1996) Designing a neural network for forecasting financial and economic time series. *Neurocomputing,* 10(3),215-236.

[14]. Brock, W. and Lakonishok, J. and Lebaron, B. (1992) Simple Technical Trading Rules and the Stochastic Properties of Stock Returns. *The Journal of Finance, 47(5), 1731-1764.*

[15]. Brooks, C. (2009) *Introductory Econometrics for Finance*. New York, Cambridge University Press.

[16]. Brown, D. P and Jennings, R. H. (1989) On Technical Analysis. *The Review of Financial Studies*, 4(1), 527-551.

[17]. Cardano, G. (1564) *The Book on Games of Chance*, New York, Holt, Rinehart and Winston, inc.

[18]. Chang, D. (2011) Testing Some of Benjamin Grahams' Stock Selection Criteria: A Case of the FTSE Bursa Malaysia EMAS Index From 2000 to 2009. *Journal of Management and Entrepreneurship, 13(2), 99-106.*

[19]. Charef, F. and Ayachi F. (2016) A Comparison between Neural Networks and GARCH Models in Exchange Rate Forecasting. *International Journal of Academic Research in Accounting, Finance and Management Sciences*, 6(1),94-99.

[20]. Cont R. (2001). Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance,* 1(2),223-236.

[21]. Cybenko G. (1989). Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control Signals and Systems*, 2(4), 303-314.

[22]. De Bondt W. and Thaler R. (1985) Does the Stock Market Overreact. *The Journal of Finance,* 40(3), 793-805.

[23]. Di Persio, L. and Honchar, O. (2017) Recurrent Neural Networks Approach to the Financial Forecast of Google Assets. *International Journal of Mathematics and Computers in Simulation*, 11, 7-13.

[24]. Di Persio, L. and Honchar, O. (2016). Artificial Neural Networks architectures for stock price prediction: comparisons and applications. *International Journal of Circuits, Systems and Signal Processing*, *10*, 403-413.

[25]. Dorffner, G. (1996) Neural Networks for Time Series. *Neural Networks World, 6(4), 447-468.*

[26]. Duchi, J. and Hazan, E. and Singer, Y. (2011) Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12(2), 121–2159.

[27]. Dunis, C. and Williams, M. (2002) "Modeling and trading the Euro/Us Dollar exchange rate: do neural networks perform better?", *Derivatives Use, Trading and Regulation*, 8 (3),211-240.

[28]. Dunis, C. L. and Karathanasopoulos A. and Laws J. (2012) Modelling and Trading the Greek Stock Market with Hybrid ARMA-Neural Network Models. *Springer Optimization and its Applications,* 70, 103-127.

[29]. Eck and J. Schmidhuber (2002) Learning the long-term structure of the blues. *In Proc. of ICANN*, 284-289.

[30]. Elton, E.J. and Gruber, M.J. and Blake, C.R. (1996) Survivorship Bias and Mutual Fund Performance. *Review of Financial Studies, 9(4), 1097-1120.*

[31]. Enke, D. and Thawornwong, S. (2005) The use of data mining and neural networks for forecasting stock market returns. *Expert Systems with Applications,* 29(4), 927-940.

[32]. Fadlalla, A. and Lin, C. (2001). An Analysis of the Applications of Neural Networks in Finance. *Interfaces*, 31(4),112-122.

[33]. Fama E.F. and Blume M. E. (1966) Filter Rules and Stock Market Trading. *Journal of Business, 39(1), 226-241.*

[34]. Fama, E.F. (1970) Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, *25*(2),383-417.

[35]. Fama, E.F. (1991) Efficient Capital Markets: II. *The journal of finance, 46*(5),1575-1617.

[36]. Fama, E.F. and French, K.R (2004) The Capital Asset Pricing Model: Theory and Evidence. *The Journal of Economic Perspectives*, 18(3), 25-46.

[37]. Fausett, L.V. (1994) *Fundamentals of neural networks*, New Jersey, Prentice-Hall.

[38]. Fassas, A. (2010) Mispricing in Stock Index Futures Markets-The CaseofGreece.[online].Available:http://papers.ssrn.com/sol3/papers.cfm?abstract_id=18 [Accessed 19th January 2018]

[39]. Fischer, T. and Krauss, C. (2017) *Deep learning with long short-term memory networks for financial market predictions* (No. 11/2017). FAU Discussion Papers in Economics.

[40]. Galagedera, D.U. (2007) A Review of Capital Asset Pricing Models. *Managerial Finance, 33(10), 821-832.*

[41]. Garcia, M.J.R. (2013) Financial Education and Behavioral Finance: New Insights into the Role of Information in Financial Decisions. *Journal of Economic Surveys,27(2), 297-315.*

[42]. Gers, F. A. and Eck, D. and Schmidhuber, J. (2002) Applying LSTM to Time Series Predictable Through Time-Window Approaches. *Neural Nets WIRN* Vietri-01. *Perspectives in Neural Computing. Springer, London*

[43]. Ghiassi, M. and Saidane, H. and Zimbra, D. K. (2005) A dynamic artificial neural network model for forecasting series events, *International Journal of Forecasting*, 21(2), 341–62.

[44]. Gilson, R. J. and Reinier H. (1984) The Mechanisms of Market Efficiency. *Virginia Law Review,* 70(4), 549-644.

[45]. Goldsborough, P. (2016) A Tour of TensorFlow. *arXiv preprint arXiv:1610.01178*.

[46]. Graves, A. and Schmidhuber, J. (2005) Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks, 18(5), 602-610.*

[47]. Graves, A. and Mohamed, A. and Hinton, G. (2013) Speech recognition with deep recurrent neural networks. *In Proc. of ICASSP*, 6645-6649.

[48]. Graham, B. (1973) *The Intelligent Investor*. New York, Harper Collins Books.

[49]. Guresen, E. and Kayakutlu G. and Daim T. U. (2011) Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389-10397.

[50]. Habib, A. and Hamid, A. S. (2014) Financial Forecasting with Neural Networks. *Academy of Accounting and Financial Studies Journal,* 18(4),37-55.

[51]. Haykin S. (1998) *Neural Networks: A Comprehensive Foundation. Ontario,* Pearson Education.

[52]. Hilovska, K. and Koncz, P. (2012) Application of Artificial Intelligence and Data Mining Techniques to Financial Markets. *Acta všfs*, 6(3), 62-74.

[53]. Hilpisch, Y. (2014) *Python for Finance: Analyze Big Financial Data. Sebastopol,* O'Reilly Media, Inc.

[54]. Hull, C. J. (2008) *Options, futures and other derivatives.* U.S.A, Pearson Education.

[55]. Hochreiter, S. and Schmidhuber, J. (1997) Long short-term memory. *Neural Computation,* 9(8),1735-1780.

[56]. Hodnett, K. and Hsieh, H.H. (2012) Capital MarketTheories: Market Efficiency Versus Investor Prospects. *The international Bussiness and Economic Research Journa*l,11(8), 849-862.

[57]. Hornik K. (1991). Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks*, 4(2),251-257.

[58]. Investing. (2017) Greece 20 Futures Chart. [online]. Available: https://www.investing.com/indices/greece-20-chart [Accessed 15th December 2017]

[59]. Jacobs, B.I., and Levy, K. N. (2006) Enhanced active equity strategies. *The Journal of Portfolio Management*, 32(3),45-55.

[60]. Johnson D. J. and Whinston B. A. (1994). *Advances in Artificial Intelligence in Economics, Finance and Management.* Greenwich, JAI Press.

[61]. Kahneman, D., and Tversky, A. (1973). On the psychology of prediction. *Psychological Review*, 80(4), 237-251.

[62]. Kavajecz, A. K. and Odders, E. (2004) Technical Analysis and Liquidity Provision. *Review of Financial Studies*, 17(4), 1043-1071.

[63]. Kingma, D. P. Ba, J. (2014) Adam: A Method for Stochastic Optimization. Proceedings of the 3rd International Conference on Learning Representations (ICLR)

[64]. Konstandinidis, A. and Katarachia, A. and Borovas, G. and Voutsa, M.E. (2012) From Efficient Market Hypothesis to Behavioral Finance: Can Behavioral Finance Be the New Dominant Model for Investing. *Scientific Bulletin-Economic Sciences,*11(2), 16-26.

[65]. Kobayashi, S. and Shirayama, S. (2017). Time Series Forecasting with Multiple Deep Learners: Selection from a Bayesian Network. *Journal of Data Analysis and Information Processing*, 5(03),115-130.

[66]. Kouretas, G. P. and Vlamis, P. (2010). The Greek crisis: Causes and Implications. *Panoeconomicus, 57(4), 391-404.*

[67]. Kumar, D. and Murugan, S. (2013). Performance analysis of Indian stock market index using neural network time series model. *International conference on pattern recognition, informatics and mobile engineering* (pp. 72–78). IEEE.

[68]. Kurihara, Y. and Nezu, E. (2010) Relationship between Japanese Stock Market Prices and Macroeconomic Variables during Quantitative Easing Period. *Briefing Notes in Economics*, 82, 8-15.

[69]. Kuo, R.J. and Chen, C.H and Hwang, Y.C. (2001) An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network. *Fuzzy Sets and Systems*, 118(1), 21-45.

[70]. Kutty, G. (2010) The Relationship Between Exchange Rates and Stock Prices: The case of Mexico. *North American Journal of Finance and Banking Research, 4(4), 1-12.*

[71]. Laopodis, N. (2012) *Understanding investments: theories and strategies.* New York, Routledge.

[72]. Levy, H. and Post, T. (2005) *Investments.* Essex, Pearson Education Limited.

[73]. Lin, C. S. and Chiu, S. H. and Lin, T. Y. (2012). Empirical mode decomposition–based least squares support vector regression for foreign exchange rate forecasting. *Economic Modelling, 29*(6), 2583–2590.

[74]. Liu, F. and Wang, J. (2012). Fluctuation prediction of stock market index by Legendre neural network with random time strength function. *Neurocomputing, 83,* 12–21.

[75]. Lu, C. J. and Lee, T. S. and Chiu, C. C. (2009). Financial time series forecasting using in-dependent component analysis and support vector regression. *Decision Support Systems,* 47(2), 115–125.

[76]. Lukac, L. P. and Bronsen. B. W. (1990). The usefulness of Historical Data in Selecting Parameters for Technical Trading Systems. *Journal of Futures Markets,* 9(1), 55-65.

[77]. Markowitz, H. (1952). Portfolio Selection. *The Journal of Finance,* 7(1), 77-91.

[78]. McCulloch S. W. and Pitts W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics,* 5(4), 115-133.

[79]. Minahan, J. R. (2006). The Role of investment Philosophy in Evaluating Investment Managers. *The Journal of Investing,* 15(2), 6-11.

[80]. Minsky, M. and Papert, S. (1969). *Perceptrons: an introduction to computational geometry*. Cambridge, M.I.T press.

[81]. Mullins, D.W. (1982). Does the Capital Asset Pricing Model Work? *Harvard Business Review,* 60(1), 105-114.

[82]. Narayan, P.K. and Narayan, S. and Singh, H. (2014) The Determinants of Stock Prices: New Evidence from the Indian banking sector. *Emerging Banking Finance and Trade*, 50(2), 5-15.

[83]. Neftci, S. N. and Policano A. J. (1984). Can Chartists Outperform the Market? Market Efficiency Tests for Technical Analysis. *Journal of Futures Markets,* 4(4), 465-478.

[84]. Nelson, D.M. and Pereira, A.C. and De Oliveira, R.A. (2017). Stock market's price movement prediction with LSTM neural networks. In *Neural Networks (IJCNN), 2017 International Joint Conference on*, 1419-1426.

[85]. Numerai (2017) A new kind of hedge fund built by a network of data scientists. [online]. Available: https://numer.ai/ [Accessed 10th October 2017]

[86]. Oppenheimer H. R. and Schlarbaum. G. G. (1981). Investing with Ben Graham: An Ex Ante Test if the Efficient Market Hypothesis: An Ex Ante Test of the Efficient Market Hypothesis. *The Journal of Financial and Quantitative Analysis,* 16(3), 341-360.

[87]. Oppenheimer H. R. and Sclarbaum. G. G. (1983). Investment Policies of Property-Liability Insurers and Pension Plans: A lessons from Ben Graham. *The Journal of Risk and Insurance,* 50(4), 611-630.

[88]. Oppenheimer H. R. (1984). A test of Ben Graham's Stock Selection Criteria. *Financial Analysis Journal,* 40(50), 68-74.

[89]. Pascanu, R., Mikolov, T. and Bengio, Y (2013). On the difficulty of training recurrent neural networks. *In Proceedings of the 30th International Conference on International Conference on Machine Learning-Volume 28,* 1310-1318.

[90]. Patra, T. and Poshakwale, S. (2006). Economic Variables and Stock Market Returns: Evidence from the Athens Stock Exchange. *Applied Financial Economics*, 16(13), 993-1005.

[91]. Perold, A.F. (2004). The Capital Asset Pricing Model. *The Journal of Economic Perspectives,* 18(3), 3-24.

[92]. Perren, M. and Faseruk, A. and Cooper, T. (2015) Making Sense of Behavioral Finance. *The Journal of Business Diversity,* 15(1), 14-22.

[93]. Pring, M. J. (2014). *Technical Analysis Explained. U.S.A*, McGraw-Hill Education.

[94]. Qi, M. and Wu, Y. (2006). Technical Trading-Rule Profitability, Data Snooping and Reality Check: Evidence from the Foreign Exchange Market. *Journal of Money, Credit and Banking, 38(8), 2135-2158.*

[95]. Quantstart. (2017). QsTrader – Free open source backtesting and live trading software. [online]. Available: https://www.quantstart.com/qstrader [Accessed 12th December 2017]

[96]. Quantopian. (2017). Quantopian overview. [online]. Available: https://www.quantopian.com/help [Accessed 10 October 2017]

[97]. Reilly, F.K and Brown, K.C (2012). *Investment Analysis and Portfolio Management.* USA. South Western.

[98]. Rosenblatt, F. (1967). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Washington, D.C, Spartan Books.

[99]. Ricciardi, V. and Simon, H.K. (2000). What is behavioral finance. *Business Education and Technology Journal,* 2(2), 1-9.

[100]. Rumelhart, D., Hinton, G. E. and Williams. R. J. (1986). Learning representation by back-propagating errors. *Nature*, 323, 533-536.

[101]. S&P Dow Jones Indices. (2017) S&P 500 (SPX). [online]. Available: http://www.us.spindices.com/indices/equity/sp-500/ [Accessed 5th May 2017]

[102]. Schaede, U. (1989). Forwards and futures in Tokugawa-period Japan: A new perspective on the Dōjima rice market. *Journal of Banking & Finance,* 13(4), 487-513.

[103]. Sewell M. (2011) History of the Efficient Market Hypothesis. *UCL Department of Computer Science,* Research Note. RN/11/04.[online].Availiable:http://www.cs.ucl.ac.uk/fileadmin/UCLCS/images/Research_Student_Information/RN_11_04.pdf [Accesed 10th October 2017]

[104]. Shankar, D. and Dhankar, R.S. (2015). Understanding the Behavior of Stock Market Functionality: Need and Role of Behavioral Finance. *Review of Management*, 5(3/4), 5-11.

[105]. Shapiro, A. F. (2000) A hitchhiker's guide to the techniques of adaptive nonlinear models. *Insurance, Mathematics and Economics,* 26(2), 119-32.

[106]. Sharpe W. F. (1994). The Sharpe Ratio. *Journal of Portfolio Management,* 21(1), 49-58.

[107]. Si, Y. W. and Yin, J. (2013). OBST-based segmentation approach to financial time series. *Engineering Applications of Artificial Intelligence,* 26 (10), 2581–2596.

[108]. Singh, J. and Kaur, K. (2014). Examining the relevance of Grahams Criteria on Indian Stock Market. *Journal of Advances in Management Research,* 11(3),273-289.

[109]. Sonoda S. and Murata N. (2017). Neural Network with Unbounded Activation Function is Universal Approximator. *Applied and Computational Harmonic Analysis,* 43(2), 233-268.

[110]. Sohail, N. and Husain, Z. (2009). Long Run and Short Run Relationships Between Macroeconomic Variables and Stock Prices in Pakistan: The case of Lahore Stock Exchange. *Pakistan Economic and Social Review*, 47(2), 183-198.

[111]. Srivastava, N. and Hinton, G.E. and Krizhevsky, A. and Sutskever, I. and Salakhutdinov, R. (2014) Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research, 15*(1), 1929-1958.

[112]. Stegemann, J.A. and Buenfeld, N. R. (1999) A Glossary of Basic Neural Networks Terminology for Regression Problems. *Neural Computing and Applications,* 8(4), 290-296.

[113]. Stout, L.A. (2003) The Mechanics of Markets Inefficiencies: An Introduction to the New Finance. *Journal of Corporation Finance,* 28(4), 635-69.

[114]. Stol H. R. and Whayley E. R. (1990). The Dynamics of Stock index and Stock Index Futures returns. *Journal of Financial and Quantitative Analysis.*25(4), 441-468.

[115]. Sundermeyer, M. and Schluter, R. and Ney, H. (2012) LSTM neural networks for language modeling.[online] Available: https://pdfs.semanticscholar.org/f9a1/b3850dfd837793743565a8af95973d395a4e.pdf [Accessed 22 January 2018].

[116]. Sutskever, I. and Vinyals, O. and Le, Q. V. (2014) Sequence to sequence learning with neural networks. *Proceedings of the 27th International Conference on Neural Information Processing Systems, Cambridge, USA, MIT Press,* 3104-3112.

[117]. Tapa A. and Yean S.C and Ahmad S. N. (2016) Modified Moving-average Crossover Trading Strategy: Evidence in Malaysia Equity Market. *International Journal of Economics and Financial Issues*, 6(S7), 149-153.

[118]. Tay, F. E., & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *Omega,* 29(4), 309–317.

[119]. Teixeira, L. A., & Oliveira, A. L. I. d. (2010). A method for automatic stock trading combining technical analysis and nearest neighbor classification. *Expert Systems with Applications,* 37(10), 6885–6890.

[120]. Theofilatos, k. and Likothanasis, S. and Karathanasopoulos, A. (2012). Modeling and Trading the EUR/USD Exchange Rate Using Machine Learning Techniques. *Engineering, Technology and Applied Science Research,* 2(5), 269-272.

[121]. Tieleman, T. and Hinton, G. (2012) Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude COURSERA: Neural networks for machine learning 4 (2), 26-31.

[122]. Tsaih R. and Hsu Y. and Lai C. (1998). Forecasting S&P 500 stock index futures with a hybrid AI system. *Decision Support Systems,* 23(2),161-174.

[123]. Twomey, J.M. and Smith, A.E (1995) Performance measures, consistency and power for artificial neural network models. *Mathematical and Computer Modeling*, 21(2), 243-258.

[124]. Varia J, and Mathew S. (2014) Overview of Amazon Web Services. [online].Available:http://emacromail,com/techpapers/Overview%20of%20Amazon%20Web%20Services.pdf [Accesed 10th December 2017]

[125]. Vasilakis, G. and Theofilatos, K. and Georgopoulos, E. and Karathanasopoulos, S. and Likothanasis, S. (2012). A Genetic Programming Approach for Eur/Usd Exchange Rate Forecasting and Trading. *Computational Economics*, *42*(4), pp.415-431.

[126]. Vinials and Le (2014) Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems*, 3104-3112.

[127]. Wang, J. Z. and Wang, J. J. and Zhang, Z. G. (2011) Forecasting stock indices with backpropagation neural network. *Expert Systems with Applications*, 38(11),14346-14355.

[128]. Wei, H. and Kin, K. L. and Nakamori Y. and Shouyang, W. and Lean, Yu. (2007) Neural Networks in Finance and Economics Forecasting. *International Journal of Information Technology & Decision Making,* 6(1), 113-140.

[129]. Wollmer, M. and Kaiser, M. and Eyben, F. and Weninger F. and Schuller B. and Rigoll G. (2012) Fully automatic audiovisual emotion recognition-voice, words, and the face. In Proceedings of Speech Communication; 10. ITG Symposium, 1-4.

[130]. Wu, C. and Luo, P. and Li, Y. and Wang, L. and Chen, K. (2015). Stock Price Forecasting: Hybrid Model of Artificial Intelligent Methods. *Inzinerine Ekonomika-Engineering Economics*, 26(1), 40-48.

[131]. Ye, Y. (2013). Application of the Stock Selection Criteria of Three Value Investors, Benjamin Graham, Peter Lynch and Joel Greenblatt: A Case of Shanghai Stock Exchange from 2006 to 2011. *International Journal of Scientific and Research Publications, 3(8),* 2250-3153.

[132]. Yoo, P. D. and Kim, M. H. and Jan, T. (2005). Machine learning techniques and use of event information for stock market prediction: A survey and evaluation. *Computational intelligence for modeling, control and automation and international conference on intelligent agents, web technologies and internet commerce*, 835–841.

[133]. Zelle, J.M. (2004) *Python programming: an introduction to computer science*. USA. Franklin, Beedle & Associates, Inc.

[134]. Zhang, G. and Patuwo, E. P. and Hu, Y. M. (1998) Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14(1), 35-62.

[135]. Zhang, Y. and Wu, L. (2009) Stock markets prediction of S&P 500 via combination of improved BCO approach and BP neural network. *Expert Systems with Applications,* 36(5), 8849-8854.

Appendices

Appendix A. The Multilayered perceptron code

```
import numpy
import matplotlib.pyplot as plt
from pandas import read_csv
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from sklearn.preprocessing import MinMaxScaler
# convert an array of values into a dataset matrix
def create_dataset(dataset, look_back=1):
	dataX, dataY = [], []
	for i in range(len(dataset)-look_back-1):
		a = dataset[i:(i+look_back), 0]
		dataX.append(a)
		dataY.append(dataset[i + look_back, 0])
	return numpy.array(dataX), numpy.array(dataY)

# fix random seed for reproducibility
numpy.random.seed(7)
```

```
# load the dataset
dataframe = read_csv('2001-2017VERYNEW2.csv', usecols=[1], engine='python', skipfooter=3)
dataset = dataframe.values
dataset = dataset.astype('float32')

# normalize the dataset
scaler = MinMaxScaler(feature_range=(-1, 1))
dataset = scaler.fit_transform(dataset)

# split into train and test sets
train_size = int(len(dataset) * 0.67)
test_size = len(dataset) - train_size
train, test = dataset[0:train_size,:], dataset[train_size:len(dataset),:]

# reshape dataset
look_back = 3
trainX, trainY = create_dataset(train, look_back)
testX, testY = create_dataset(test, look_back)
```

```
# create and fit Multilayer Perceptron model
model = Sequential()
model.add(Dense(24, input_dim=look_back, activation='relu'))
model.add(Dense(12, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam')
model.fit(trainX, trainY, epochs=100, batch_size=2, verbose=2)
# Estimate model performance
trainScore = model.evaluate(trainX, trainY, verbose=0)
print('Train Score: %.2f MSE (%.2f RMSE)' % (trainScore, math.sqrt(trainScore)))
testScore = model.evaluate(testX, testY, verbose=0)
print('Test Score: %.2f MSE (%.2f RMSE)' % (testScore, math.sqrt(testScore)))
# save model to single file
model.save('MLPWINDOW.h5')
# generate predictions for training
trainPredict = model.predict(trainX)
testPredict = model.predict(testX)
```

```
# shift train predictions for plotting
trainPredictPlot = numpy.empty_like(dataset)
trainPredictPlot[:, :] = numpy.nan
trainPredictPlot[look_back:len(trainPredict)+look_back, :] = trainPredict

# shift test predictions for plotting
testPredictPlot = numpy.empty_like(dataset)
testPredictPlot[:, :] = numpy.nan
testPredictPlot[len(trainPredict)+(look_back*2)+1:len(dataset)-1, :] = testPredict

# plot baseline and predictions
plt.plot(dataset)
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
```

Appendix B. The Long-Short Term Memory Network code

```
# Stacked LSTM for Athens stock exchange index forecast with memory
import numpy
import matplotlib.pyplot as plt
from pandas import read_csv
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error


# convert an array of values into a dataset matrix
def create_dataset(dataset, look_back=1):
        dataX, dataY = [], []
        for i in range(len(dataset)-look_back-1):
                a = dataset[i:(i+look_back), 0]
                dataX.append(a)
                dataY.append(dataset[i + look_back, 0])
        return numpy.array(dataX), numpy.array(dataY)
```

```
# fix random seed for reproducibility

numpy.random.seed(7)

# load the dataset

dataframe = read_csv('2001-2017VERYNEW2.csv', usecols=[1], engine='python', skipfooter=3)

dataset = dataframe.values

dataset = dataset.astype('float32')


# normalize the dataset

scaler = MinMaxScaler(feature_range=(-1, 1))

dataset = scaler.fit_transform(dataset)


# split into train and test sets

train_size = int(len(dataset) * 0.67)

test_size = len(dataset) - train_size

train, test = dataset[0:train_size,:], dataset[train_size:len(dataset),:]


# reshape into X=t and Y=t+1

look_back = 3

trainX, trainY = create_dataset(train, look_back)

testX, testY = create_dataset(test, look_back)
```

```
# reshape input to be [samples, time steps, features]
trainX = numpy.reshape(trainX, (trainX.shape[0], trainX.shape[1], 1))
testX = numpy.reshape(testX, (testX.shape[0], testX.shape[1], 1))


# create and fit the LSTM network
batch_size = 1
model = Sequential()
model.add(LSTM(4, batch_input_shape=(batch_size, look_back, 1), stateful=True, return_sequences=True))
model.add(LSTM(50, batch_input_shape=(batch_size, look_back, 1), stateful=True))
model.add(Dropout(0.2))
model.add(LSTM(100, batch_input_shape=(batch_size, look_back, 1), stateful=True))
model.add(Dropout(0.2))
model.add(Dense(1))
model.compile(loss='mse', optimizer='adam')
model.fit(trainX, trainY, epochs=100, batch_size=batch_size, verbose=2, shuffle=False)
model.reset_states()
# make predictions
trainPredict = model.predict(trainX, batch_size=batch_size)
model.reset_states()
```

```
# invert predictions
trainPredict = scaler.inverse_transform(trainPredict)
trainY = scaler.inverse_transform([trainY])
testPredict = scaler.inverse_transform(testPredict)
testY = scaler.inverse_transform([testY])

# calculate root mean squared error
trainScore = math.sqrt(mean_squared_error(trainY[0], trainPredict[:,0]))
print('Train Score: %.2f RMSE' % (trainScore))
testScore = math.sqrt(mean_squared_error(testY[0], testPredict[:,0]))
print('Test Score: %.2f RMSE' % (testScore))

# shift train predictions for plotting
trainPredictPlot = numpy.empty_like(dataset)
trainPredictPlot[:, :] = numpy.nan
trainPredictPlot[look_back:len(trainPredict)+look_back, :] = trainPredict

# shift test predictions for plotting
testPredictPlot = numpy.empty_like(dataset)
testPredictPlot[:, :] = numpy.nan
```

```
testPredictPlot[len(trainPredict)+(look_back*2)+1:len(dataset)-1, :] = testPredict


# plot baseline and predictions
plt.plot(scaler.inverse_transform(dataset))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
```

Appendix C. The ARMA model Residuals