Using Long Short-Term Memory to predict stock price movement for S&P500

Andrea Bugeja
University of Malta

***Abstract*—It is well knowledge that past stock market performance is not reproducible or indicative of future performance. In this situation, however, the research will focus on how to develop a machine learning model to forecast future stock price movement. The reader will gain a better grasp of machine learning and how it is implemented in the financial area in the following sections. Various literatures are chosen to provide a better context on how the described method is used and afterwards implemented. Finally, the method's application will be explained, and the results and outcomes will be provided.**

Keywords—machine learning, LSTM, Algorithmic Trading, Finance, Artificial Intelligence

I. Introduction

This section will provide a brief review of algorithmic trading before moving on to the challenge encountered in the field and what it includes.

Technically, the market can be characterised and understood as a set of sinusoidal functions that alternate between two states: uptrend, also known as a bull market and downtrend, also known as a bear market. This sinusoidal wave is then fed into computer programmes that execute on a predefined set of instructions and are used to place an order/transaction. Sets of instructions are defined using timing, pricing, quantity, or any mathematical model. This is referred to as algorithmic trading [1].

Algorithmic trading comes with many advantages compared to human traders, some of which include:

- Fast opening/closing of orders resulting in competitive pricing
- Degree of precision when creating orders
- Ability to check multiple markets at the same time
- Faster interpretation of data
- Better pattern recognition
- The removal of human emotions and psychological considerations reduce the probability of errors present during trading.

Aside from profit prospects the advantages of algorithmic trading are obvious, algo-trading makes markets more liquid and trading more methodical by removing the influence of human emotions on trading activity, while also achieving the best execution of trades[1].

Volatility [2], nonstationary [3], periodicity [4], nonlinearity [5], and long-term dependence [6] are all common properties of financial time series. Traditional statistical models may capture the volatility and periodicity of financial series accurately. However, they have difficulty analysing non-stationary sequences and capturing the nonlinear relationships of financial time series.

Traditional statistical models, such as the multiple regression method, the Autoregressive integrated moving average (ARIMA) model [7], and the Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model [8], are widely used in the real world and can accurately capture the volatility and periodicity of financial series. The statistical models are also quite easy to understand. However, they have difficulty analysing non-stationary sequences and capturing the nonlinear relationships of financial time series [9].

A neural-network technique, comprising a Recurrent Neural Network (RNN) [10], can be used to cope with non-stationary sequences while simultaneously capturing non-linear interactions. Traditional RNN models, on the other hand, have another challenge: the vanishing gradient problem, even though only a few of them can accurately reflect a time series' long-term dependence [10]. Furthermore, because raw timeseries is frequently utilised as input in neural networks, describing the input properties picked in predictions is problematic.

To minimize the vanishing gradient issue in RNNs while also overcoming the long-term dependence in predictions, an LSTM model will be used [11, p.]. LSTM is a variation of the RNN architecture; however, it incorporates a processor known as memory cell into the algorithm to help determine whether the information is valuable.

There are three gates in a memory cell: a forget gate, an input gate, and an output gate. The rules are applied to a message as it enters the LSTM network. Only information that matches the algorithm's certification is kept, while information that does not match is destroyed using the forget gate. This enables a clear depiction of the long-term reliance of the financial time series.

The contribution being made here is to present an alternative machine learning model to that of the traditional RNN, while also aiming to achieve better results for LSTM and ARIMA than those in [12] .

II. Literature Review

This section will provide a brief overview of the algorithmic trading literature, where we will look at some of the machine learning models that are used and the

algorithmic trading research that focuses on price prediction.

A. Autoregressive Integrated Moving Average

ARIMA processes have been extensively studied and are a critical component of time series analysis. George Box and Gwilym Jenkins popularised them in the early 1970s, and as a result, ARIMA processes are sometimes referred to as Box-Jenkins models [13].

The ARIMA forecasting approach is based on the following ideas:

- Forecasts are based on linear timeseries [14].
- The goal is to create models with as few parameters as possible that provide a sufficient description of the observed data. This is referred to as the principle of parsimony [14].

The ARIMA Model is a generalised model of the Autoregressive Moving Average (ARMA) that integrates the Autoregressive (AR) and Moving Average (MA) processes to create a time series composite model. ARIMA(p, d, q) captures the model's major elements, as the acronym suggests [12]:

- Autoregressive: A regression model that considers the relationships between an observation and several lagged observations (p)

Equation 1: Autoregressive Model Equation

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} + \epsilon_1$$

- Integrated: To make the time series stable, measure the differences between observations made at different times (d).
- Moving Average: When applying a moving average model to lagged observations, this method considers the dependency between observations and the residual error components (q).

Equation 2: Moving Average Model Equation

$$Y_t = \alpha + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q}$$

B. LSTM

As briefly mentioned in Section 1, LSTM is a form of RNN that can remember data from prior stages to use it in the future [10]. However, before delving deeper into LSTM, it is vital to have a basic grasp of the network itself.

1) Artificial Neural Network (ANN)

An ANN has at least three layers: an input layer, a hidden layer, and an output layer. The number of nodes in the input layer is governed by the characteristics chosen during data collection, which are then linked to the nodes in the hidden layer via "synapses." Weights passed through synaptic linkages operate as a decision maker in determining whether a signal, or input, may or may not pass through [15].

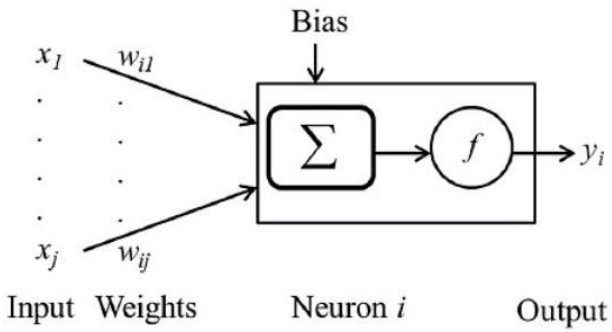

Figure 1: Simple overview of ANN

Nodes are computer programme layers that work in two ways: the output layer builds a vector of probabilities for different outcomes and chooses the one with the lowest error rate or cost. To translate the inputs to the outputs, the nodes in the hidden layers apply an activation function such as sigmoid activation or tanh activation to the weighted sum of inputs [15]. The output layer generates a vector of probabilities for different outcomes and chooses the one with the lowest error rate or cost. The gap between expected and forecasted values is the minimum error rate or cost, often known as SoftMax [12].

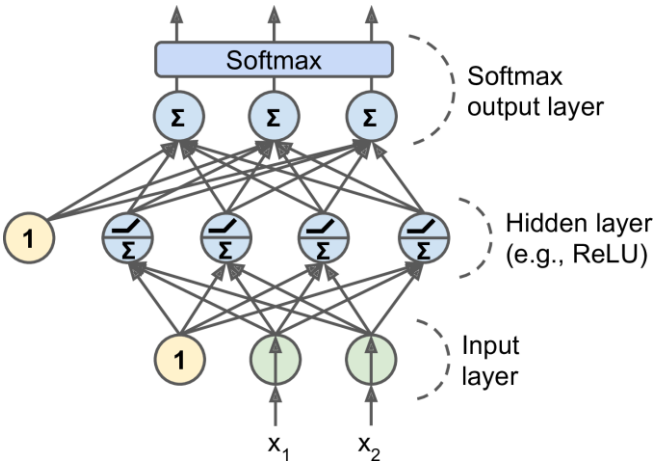

Figure 2: Detailed overview of ANN Architecture

To find the best error levels, the errors are sent back into the network from the output layer to the hidden layers, and the weights are modified as a result; this is also known as back propagation. The procedure is repeated for numerous epochs, with the weights re-adjusted until the anticipated values and, as a result, the cost improve. An epoch refers to one complete cycle. When the cost function is at its smallest, the model is considered trained [15].

2) Recurrent Neural Network (RNN)

A RNN is a form of ANN whose goal is to anticipate the next step in the sequence based on the data from the previous step [12]. RNN has found widespread application in a wide range of domains, including sequence generation [16], machine translation [17], and speech recognition [18]. In contrast to traditional deep feedback networks, RNNs have recurrent neurons that can feed back the output signal to the input. As a result, it can recall historical knowledge and learn to rely on sequential data over time. This historical knowledge is stored in the hidden layer, which act as a form of internal storage.

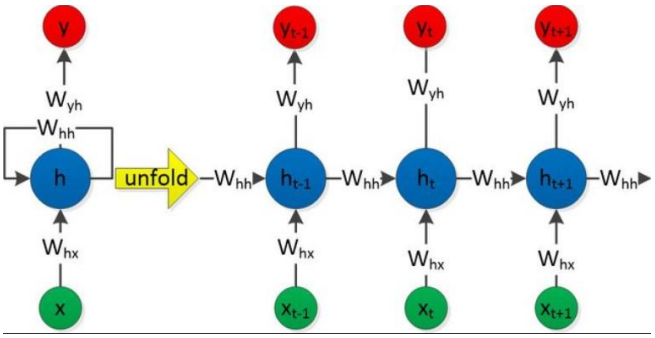

Figure 3: RNN Architecture

RNNs are called "recurrent" because they do the same task for each input item in a sequence, with the added benefit of forecasting future unobserved sequential data using previously acquired data [12]. The fundamental issue with a typical generic RNN is that these networks only remember a few past stages in the sequence and hence are insufficient for keeping longer data sequences. Unfortunately, the gradient explosion [10] or disappearance problem [9] makes the RNN architecture difficult to train; this is remedied by introducing the "memory gate" in LSTM [11].

3) Long Short-Term Memory

LSTM is a type of RNN, however as described in the preceding section, LSTM employs the memory gate. This function helps you to memorise data sequences [11]. Each LSTM is made up of cells, or system modules, that capture and store data streams.

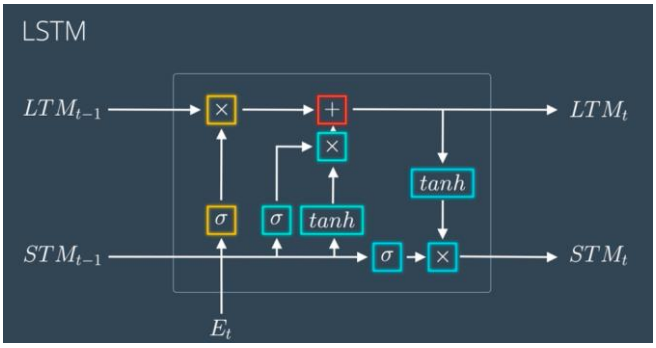

Figure 4: LSTM Architecture

The cells are like a transport line that connects one module to another, transferring data from the past and gathering it for the present. The gates, which are based on the sigmoidal neural network layer, allow cells to either allow or reject data. Each sigmoid layer produces numbers between zero and one, indicating how much of each segment of data should be allowed through in each cell. If an estimation value of 0 is achieved, data will not be allowed to pass through, however should the estimation result in a 1, all data will be allowed to pass through. A LSTM consists of three main gates, each controlling the state of each cell [10], [11]. The three gates are as follows:

- Forget Gate: A number between 1-0 is outputted, the estimation of this value defines if data will be remembered or forgotten [12].

- Memory Gate: The memory gate selects data to be stored in memory for future use. An input door layer, also known as the sigmoidal layer, determines which values to alter. A tanh layer creates a vector of fresh candidate values [12].

- Output Gate: Each cell's output is determined by this gate. The value returned is determined by the cell state, which includes both new and filtered data [12].

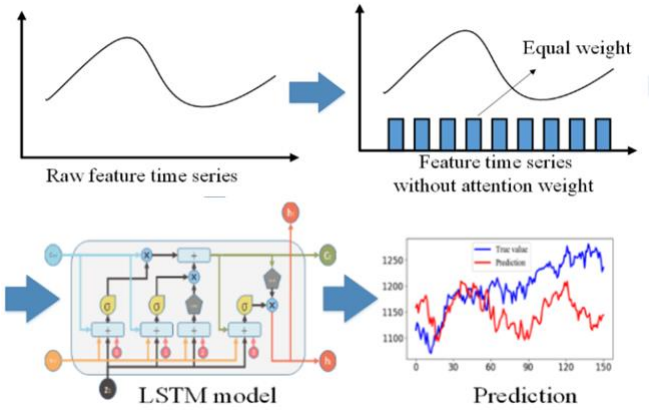

Figure 5: LSTM Architecture used

III. METHODOLOGY

In this section, three data sets are to be presented highlighting the validity of the financial time series prediction system. However, will first provide a brief overview of the dataset and experimental parameter settings before comparing it to other prediction algorithms will be discussed.

A. Datasets and Settings

The dataset selected is the SP500 index. This is an index of the top 500 best performing equities on the market; it is frequently used as a benchmark in many studies. The S&P500's price movement reflects the overall trend and direction of the market. The above dataset spans nearly 20 years, from January 1, 1999, to December 31, 2019. This

20-year dataset is divided into 80 percent training and 25% validation. The dataset range is from 01/01/2020 to 31/01/2021, this will be used for predictions. The training dataset and testing dataset are kept separate as to ensure that the model is not aware of the future price movements. The above was downloaded using the yfinance library. The yfinance library is a library which allows the downloading of historical timeseries data from yahoo finance. In this instance the following where downloaded: Open, Low, High, Close, Adj Close, Volume, however in our case Adj close will be used to train the model

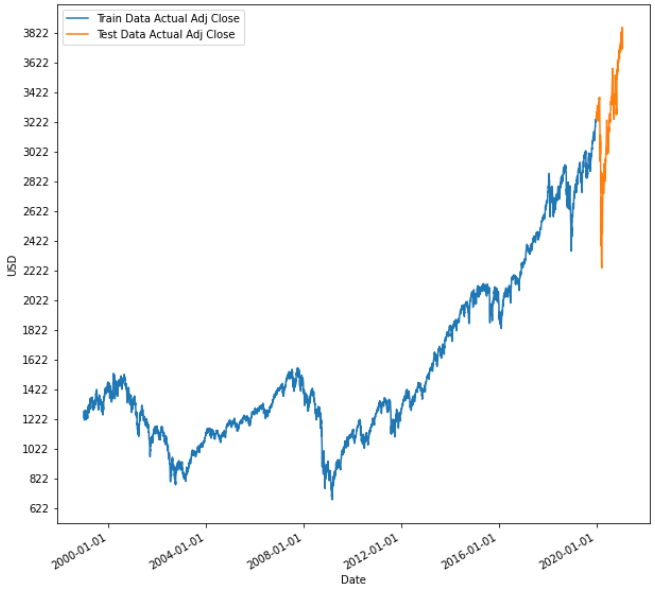

Figure 6: SP500 Training vs Validation Split

B. Validation Metric

To compare the effectiveness of different methods for financial time series prediction, Root-Mean-Square Error (RMSE) will be used as the evaluation metric. RMSE is a measure of prediction accuracy of a forecasting method in statistics. It computes the differences or residuals between the observed and predicted values. The following is the formula for calculating RMSE:

Equation 3: RMSE Formula

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}$$

In the above equation, N denotes the total number of observations, x_i is the actual value, where as $\hat{X}_i$ is the predicted value [12].

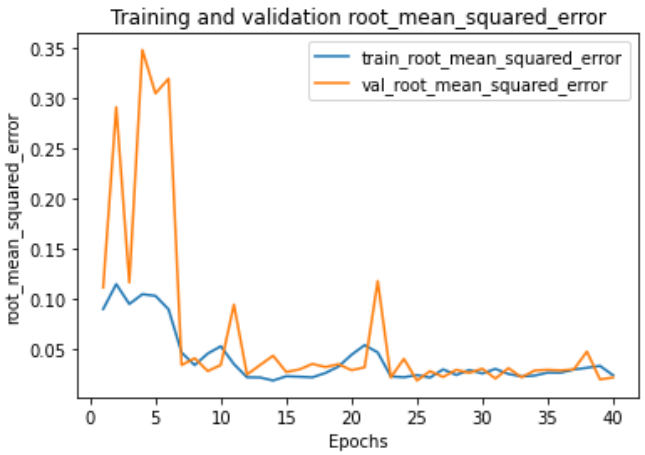

Figure 7: Training RMSE vs Validation RMSE

C. LSTM Architecture

The LSTM architecture consists of a 200-unit LSTM layer with a 0.2 rate on the dropout layer, this is repeated 4 times. The 200-unit parameter is the neuron dimension for the LSTM layer, therefore in this instance the LSTM layer has 200 hidden units or hidden cells. The dropout layer is implemented to prevent overfitting issues that can be encountered during training. During the training phase, nodes and their connections are deleted from the network at random. In our case, a 20% chance of removal was applied [19].

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_22 (LSTM)              (None, 20, 200)           161600

 dropout_22 (Dropout)        (None, 20, 200)           0

 lstm_23 (LSTM)              (None, 20, 200)           320800

 dropout_23 (Dropout)        (None, 20, 200)           0

 lstm_24 (LSTM)              (None, 20, 200)           320800

 dropout_24 (Dropout)        (None, 20, 200)           0

 lstm_25 (LSTM)              (None, 200)               320800

 dropout_25 (Dropout)        (None, 200)               0

 dense_5 (Dense)             (None, 1)                 201

=================================================================
Total params: 1,124,201
Trainable params: 1,124,201
Non-trainable params: 0
_________________________________________________________________
```

Figure 8: Model Architecture

Following, the last dropout layer, a dense layer at 1 unit is used. A dense layer is a layer that is firmly related to the layer before it and works to change the dimension of the output by executing matrix vector multiplication. Matrix vector multiplication is a method in which the row vector of the output from the preceding layers equals the column vector of the dense layer [20].

As a loss function, mean squared error (MSE) was used and while an Adam optimizer was also implemented. Furthermore, a 20-day step look back was used.

Equation 4: Mean-Squared Error Formula

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

The selection of the above architecture was mostly achieved through trial and error along with using the built in evaluate command. This allowed to vary parameters while checking for any decreases on RMSE.

As pre-processing, normalization of data was applied, scaling it between -1 and 1. Normalization is a technique that is frequently used in data preparation for machine learning. Normalisation is the process of converting the values of numeric columns in a dataset to a similar scale without distorting the ranges of values or losing information. [21].

D. Model Training

Deep learning iterative optimization algorithms frequently revolve around tuning model parameters in relation to "gradient descent." Gradient descent is coupled with a parameter called "learning rate," which represents how quickly or slowly the neural network components are learned. It is feasible to train a network more than once with the same training dataset to optimise the parameters. If the data is too large to feed into a neural network for training all at once, then might have to be split up and train the network in stages.

The parameters used for training the model where as follows, however these will be explained further below in this section:

- Batch size: 128
- Epochs: 100
- Validation split: 0.25 //25% of the training dataset will be used for validation

1) Batch Size

The total number of training data used is referred to as the batch size. Epoch, on the other hand, refers to the number of batches required to train a model utilising the entire dataset. In our case a batch size of 128 was used resulting in 31 total batches per epoch and a total of 100 epochs.

2) Epoch

The epoch represents the total number of times the dataset has been utilised for training. One epoch that the model has processed the entire dataset once. Because different datasets behave differently, different epochs may be required to train their networks best [12]. Despite setting the model to run for 100 epochs, this was training was concluded early due to the early stopping call-back in place. This call-back ensures that should there be no further improvements to the validation RMSE after 15 tries, training will be terminated. Parameters for this call-back where as follows:

- Monitor: val_root_mean_squared_error //monitor changes in validation RMSE
- Patience: 15 //Stop after 15 tries
- Mode: min //RMSE reading needs to become smaller to satisfy monitoring condition
- min_delta: 0.000001 //The minimum change that must take place to not trigger the patience parameter

IV. Results

In this section the results obtained will be presented for the SP500 will be presented and compared to ARIMA and LSTM results obtained from [12] for SP500 (GSPC).

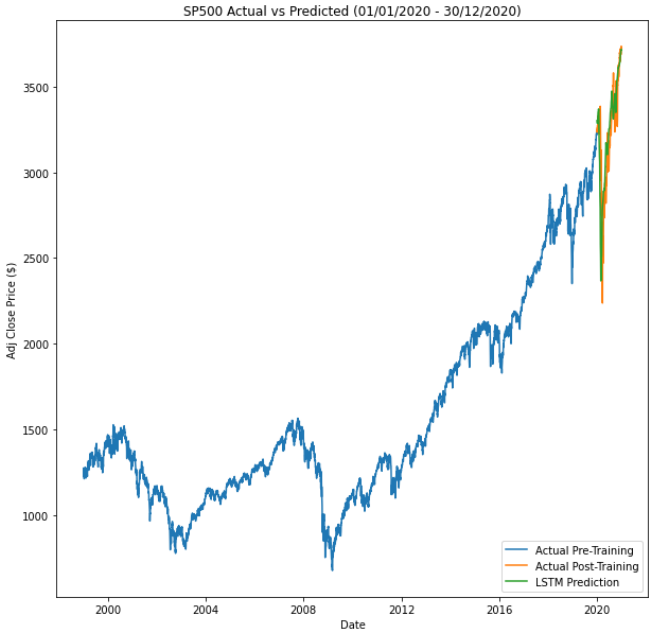

Figure 9: Graph of all actual data and LSTM prediction

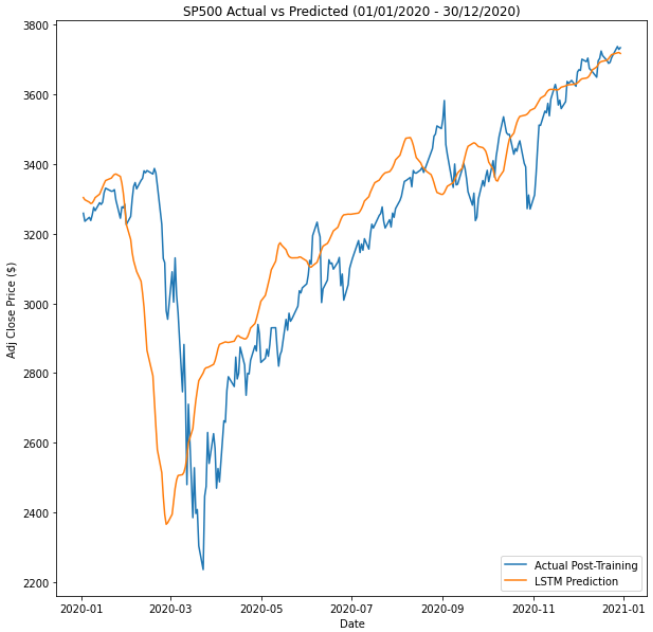

Figure 10: Graph of Actual Data vs LSTM Prediction

Table 1: LSTM metrics for S&P500 (GSPC)

Models	MSE	RMSE	MAE	RMSE % Diff
LSTM	0.0049	0.0704	0.0522	0.0%
LSTM [12]	0.0609	0.0781	N/A	-9.8591%
ARIMA [12]	0.3058	0.553	N/A	-87.2694%

The results in Table 1 show that our LSTM model has outperformed the both LSTM and ARIMA from obtained in [12] in relation to the SP500.

V. Conclusion

ARIMA and LSTM-based algorithms increased prediction by 87 percent and 9.8 percent, respectively compared to [12]. Deep learning can be used to solve a variety of additional prediction issues in finance and economics. This study argues for the advantages of applying deep learning algorithms and techniques to economic and financial data.

In terms of future work, it is planned to investigate the benefits of deep learning by applying similar techniques to new situations and datasets with varying numbers of features. Furthermore, it would be great to apply these strategies not only to well-known stocks that have a high connection to the S&P500 index, but also to other less-well-known equities. If adequate data is available, this can also be applied to the local market.

References

[1] D. S. Napate and M. Thakur, 'Algorithmic Trading and Strategies', p. 11.

[2] M. McAleer, 'Automated Inference and Learning in Modeling Financial Volatility', *Econometric Theory*, vol. 21, no. 1, pp. 232–261, 2005.

[3] R. R. Simonds, L. R. LaMotte, and A. McWhorter, 'Testing for Nonstationarity of Market: An Exact Test and Power Considerations', *The Journal of Financial and Quantitative Analysis*, vol. 21, no. 2, pp. 209–220, 1986, doi: 10.2307/2330738.

[4] T. G. Andersen and T. Bollerslev, 'Intraday periodicity and volatility persistence in financial markets', *Journal of Empirical Finance*, vol. 4, no. 2–3, pp. 115–158, Jun. 1997, doi: 10.1016/S0927-5398(97)00004-2.

[5] M. K. P. So, C. W. S. Chen, and M.-T. Chen, 'A Bayesian threshold nonlinearity test for financial time series', *J. Forecast.*, vol. 24, no. 1, pp. 61–75, Jan. 2005, doi: 10.1002/for.939.

[6] J. T. Barkoulas and C. F. Baum, 'LONG TERM DEPENDENCE IN STOCK RETURNS', p. 13, Dec. 1996.

[7] D. Asteriou and S. G. Hall, *Applied econometrics*, 2. ed. Basingstoke: Palgrave Macmillan, 2011.

[8] A. Lama, G. K. Jha, R. K. Paul, and B. Gurung, 'Modelling and Forecasting of Price Volatility: An Application of GARCH and EGARCH Models', *Agri. Econ. Rese. Revi.*, vol. 28, no. 1, p. 73, 2015, doi: 10.5958/0974-0279.2015.00005.1.

[9] X. Zhang, X. Liang, A. Zhiyuli, S. Zhang, R. Xu, and B. Wu, 'AT-LSTM: An Attention-based LSTM Model for Financial Time Series Prediction', *IOP Conf. Ser.: Mater. Sci. Eng.*, vol. 569, p. 052037, Aug. 2019, doi: 10.1088/1757-899X/569/5/052037.

[10] A. Sherstinsky, 'Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network', *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, Mar. 2020, doi: 10.1016/j.physd.2019.132306.

[11] S. Hochreiter and J. Schmidhuber, 'Long Short-term Memory', *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997, doi: 10.1162/neco.1997.9.8.1735.

[12] S. Siami-Namini, N. Tavakoli, and A. Siami Namin, 'A Comparison of ARIMA and LSTM in Forecasting Time Series', in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Orlando, FL, Dec. 2018, pp. 1394–1401. doi: 10.1109/ICMLA.2018.00227.

[13] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis*, 1st ed. Wiley, 2008. doi: 10.1002/9781118619193.

[14] R. Hyndman, 'ARIMA processes', 2001, pp. 27–28.

[15] R. Dastres and M. Soori, 'Artificial Neural Network Systems', *International Journal of Imaging and Robotics*, vol. 21, pp. 13–25, Mar. 2021.

[16] I. Sutskever, O. Vinyals, and Q. V. Le, 'Sequence to Sequence Learning with Neural Networks', in *Advances in Neural Information Processing Systems*, 2014, vol. 27. Accessed: Jun. 27, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html

[17] K. M. Hermann *et al.*, 'Teaching Machines to Read and Comprehend'. arXiv, Nov. 19, 2015. Accessed: Jun. 09, 2022. [Online]. Available: http://arxiv.org/abs/1506.03340

[18] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, 'End-to-End Attention-based Large Vocabulary Speech Recognition'. arXiv, Mar. 14, 2016. Accessed: Jun. 05, 2022. [Online]. Available: http://arxiv.org/abs/1508.04395

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 'Dropout: A Simple Way to Prevent Neural Networks from Overfitting', p. 30.

[20] Y. Verma, 'A Complete Understanding of Dense Layers in Neural Networks', *Analytics India Magazine*, Sep. 19, 2021. https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/ (accessed Jun. 06, 2022).

[21] Z. Zhang, 'Understand Data Normalization in Machine Learning', *Medium*, Aug. 11, 2019. https://towardsdatascience.com/understand-data-

normalization-in-machine-learning-8ff3062101f0
(accessed Jun. 07, 2022).

Plagiarism Declaration Form

Plagiarism is defined as "*the unacknowledged use, as one's own, of work of another person, whether or not such work has been published, and as may be further elaborated in Faculty or University guidelines*" (University Assessment Regulations, 2009, Regulation 39 (b)(i), University of Malta).

I, the undersigned, declare that the report submitted is my work, except where acknowledged and referenced. I understand that the penalties for committing a breach of the regulations include loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree programme.

Work submitted without this signed declaration will not be corrected, and will be given zero marks.

Andrea Bugeja	ARI5123	29/06/2022
Student's full name	**Study-unit code**	**Date of submission**

Title of submitted work: Using LSTM to predict price movements of S&P500

Student's Signature