Predicting and Forecasting A Global Stock Index: LSTM vs NARX

Hisho Rajanathan

Abstract This paper critically evaluates two neural network algorithms to forecast the prices of stock market indices. Long Short-Term Memory Networks (LSTM) and Nonlinear Autoregressive Network with Exogenous Inputs (NARX) will be compared. The hyperparameters for each of the neural network algorithms will be optimised using Grid Search and Bayesian Optimisation.

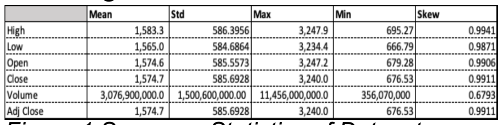
Motivation of the problem

Conventional methods of trying to beat the stock market are used by a number of traders, hedge funds and investment companies, hence a variety of different methods will need to be adopted to have a chance of outperforming the stock market to produce excess returns. Traditional approaches of technical and fundamental analysis may give investors an insight into long term investing; however using unconventional methods such as neural networks may improve chances of generating excess returns. The stock market is highly unpredictable and there has been a lot of effort to try and predict or forecast the stock market to generate excess returns [4]. Generally Artificial Neural Networks (ANN), are used for stock price prediction, time series forecasting and performance measurement. Stock prices can be seen as random time sequence with noise [5] and neural networks can find patterns best suited in environments with noise or partial information [6].

Description of the dataset

The S&P 500, a major US index, consists of 500 leading US companies which cover approximately 80% of market capitalisation [1]. The S&P 500 index values were obtained from Yahoo! Finance using the yfinance and Pandas Data Reader Library in python [2][3]. The dataset includes Opening, Closing, High, Low and Adjustment Closing prices for each day from 01/01/2000 to 01/01/2020. The stock prices do not move on weekends or public holidays in their respective country as stock markets are closed. In the original dataset, these dates were not included; hence the missing dates have been filled in using the same prices as the previous days.

From Figure 1 we can see that there is a large range for each of the prices, thus standardisation will need to be applied to each of the prices columns. For the purpose of this paper, we will only be looking at forecasting the Opening price of the S&P 500 index, due to computational resources.



	Mean	Std	Max	Min	Skew
High	1,583.3	586.3956	3,247.9	695.27	0.9941
Low	1,565.0	584.6864	3,234.4	666.79	0.9871
Open	1,574.6	585.5573	3,247.2	679.28	0.9906
Close	1,574.7	585.6928	3,240.0	676.53	0.9911
Volume	3,076,900,000.0	1,500,600,000.00	11,456,000,000.0	356,070,000	0.6793
Adj Close	1,574.7	585.6928	3,240.0	676.53	0.9911

Figure 1 Summary Statistics of Dataset

Figure 2 shows the movement of the S&P 500 index from 01/01/2000 to 01/01/2020. The index does not follow a linear relationship and has many sudden drops such as the financial crisis in 2008. The S&P 500 index suffered losses till 2009, which is when the markets slowly started to recover and there has been a general upward trend with the odd drops in price.

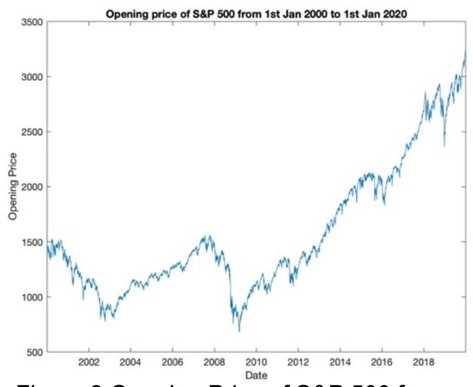


Figure 2 Opening Price of S&P 500 from 01/01/2000 to 01/01/2020

Neural network models

Long Short – Term Memory (LSTM) Network

LSTM's were introduced as an improvement for Recurrent Neural Networks (RNN), to overcome the long-term dependency problem [8]. The LSTM network trains using Backpropagation Through Time which overcomes the vanishing gradient problem [20].

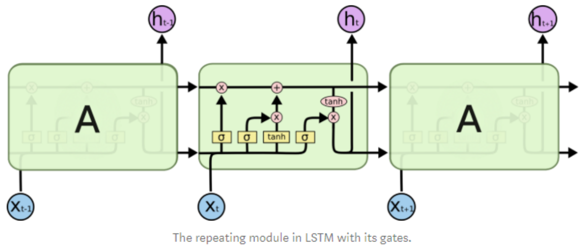


Figure 3 Example of LSTM Network [10]

Pros

LSTMs avoid the long-term dependencies that are present in RNNs, meaning that they are able to learn information for a long period.

Cons

Although the LSTM is built on the RNN, meaning it may be a better neural network for time series predictions, a disadvantage is that it requires more computational power compared to the NARX network. In addition, LSTMs require a lot more memory to train the network and are prone to overfitting.

Nonlinear Autoregressive Network with Exogenous Inputs (NARX)

The NARX model is a recurrent dynamic network, with feedback connections enclosing several layers of the network [13]. Similar to LSTM, NARX models can be used to forecast time series data and can also be used for nonlinear filtering.

Pro

NARX networks converge much faster than other neural networks and generalise much better than other networks [11]. It is also said that the gradient descent is better in NARX models compared to other networks. NARX models are also considered to be much better at determining long term dependencies compared to conventional recurrent neural networks [13].

Cons

NARX networks are prone to the vanishing gradient problem, hence are affected by the long term dependencies like conventional recurrent neural networks [18].

Hypothesis statement

This paper aims to identify the best neural network: LSTM or NARX and their optimal hyperparameters to forecast and predict the S&P 500 opening price. A couple of hypothesis statements have been created prior to creating each neural network and they are as follows:

- LSTM would have a lower RMSE, hence a higher accuracy of predicting the opening price compared to the NARX model.
- Although, LSTM may have a lower RMSE, the computational power required to run this model is a lot higher, hence the NARX model would be able to predict the opening price quicker.

Training and evaluation methodology

From training both models, 15% of the original dataset has been held out for testing to compare LSTM and NARX. 85% of the data has been used for training and validation of both models.

For the purposes of this paper, the aim is to predict the opening price of the S&P 500 index, whilst using all the inputs such as Opening Price, Closing Price etc. Part of the pre-processing steps, all the inputs had to be standardized due to the large variance in values. Cross validation has not been performed in either of the neural network models as cross validation removes the time dependent nature of the data.

Root Mean Squared Error (RMSE) and Time taken to train the models are the metric choice in comparing both models. The reasons for using time taken as one of performance measures is because it could provide the lowest RMSE, hence the best accuracy to predict the stock index. However, if it takes a long time to train meaning it will be computationally expensive.

Two methods were implemented for hyperparameter optimisation, grid search and Bayesian Optimisation. The optimal hyperparameters from both methods were fed into the final model to see which approach gives a better result to predict the Opening price. Bayesian Optimisation was another method included for hyperparameter optimisation as it is seen to be a sophisticated approach and has been shown to outperform other optimisation techniques [21].

For both models, a baseline model was created to see what the initial RMSE and time taken was and then the hyperparameters were adjusted accordingly.

The LSTM base model includes a Sequence Input Layer, LSTM Layer, Fully Connected Layer and a Regression Layer. This final layer was included as the aim is to predict and forecast the

Open Price for the S&P 500 index. The hyperparameters that were optimised were number of hidden units, learning rate, the output size for the fully connected layer and the dropout percentage for the drop out layer. The LSTM model uses the Adam optimizer which is an optimization algorithm that has been designed for training deep neural networks [17].

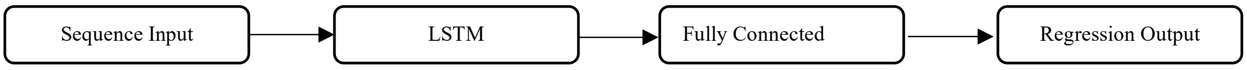


Figure 4 Initial LSTM structure

The next stage of optimisation was to add another fully connected layer and a drop out layer to the model. Adding more depth to a neural network results in better generalisation of the neural network hence another fully connected layer was added to the LSTM model [16].The number of input and output neurons were determined by the number of input and output variables; in this case the number of inputs were six and output was one. Early stopping was implemented in the LSTM model, by using Validation Patience which stops training automatically if there was no improvement over 10 iterations [22].

The NARX model was trained using a series parallel architecture, which allows the network to use real target values instead of model outputs. This has been shown to improve the accuracy of the final model and is seen as more computationally efficient [19]. Prior to testing the model, the NARX network is 'closed' to create a parallel architecture to be used for prediction as shown in the right image in Figure 5. Similarly, for the optimisation of the NARX model a grid search approach and Bayesian optimisation was used. The hyperparameters that were optimised were the input delay, feedback delay, hidden layer size and the training function. Early stopping was implemented in the NARX model, where if there is no improvement in the RMSE over 20 consecutive iterations, the network should stop training [22]. The reasoning for this is to prevent the neural network overfitting on the training data. Initially the delay size was kept the same for both the input delay and feedback delay, however varying the sizes of input delay and feedback delay found the RMSE to improve in certain scenarios.

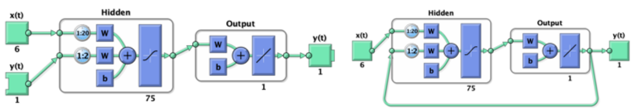


Figure 5 NARX 'open' Network (Left) NARX 'closed' Network (Right)

In order to attain the best hyperparameters for both models using a grid search, 3D plots were created to identify where the RMSE and training time was the lowest. The same
fine tuning of the hyperparameters were run multiple times to determine an average RMSE and time for each iteration of the neural network, as the RMSE scores and time varied for each run. For each of the hyperparameters that were being optimised, the models were run three times and an average was taken.

Results, Findings and Evaluation

LSTM

The optimal hyperparameters for the LSTM model using grid search were 100 Epochs, 400 Hidden Units, 0.001 Learning rate. In addition an additional fully connected layer with an output size of 100 and drop out layer with a percentage of 0.1, was included on top of the base line model which improved the RMSE score, however it did increase the calculation time significantly.

Using the structure from Figure 4 the hidden units were optimised. From Figure 6, the optimal hidden units were 400 however the calculation time drastically increased for

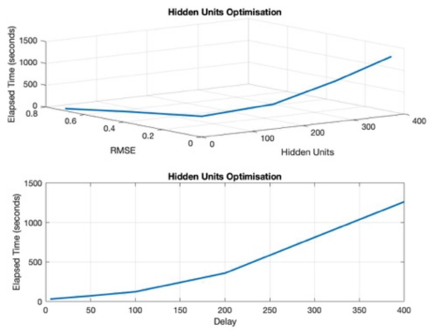


Figure 6 LSTM Grid Search: Hidden Units Optimisation

this size of hidden units. A test prior to this was conducted and it showed that increasing the hidden units size, improved the RMSE minimally however the calculation time increased. The RMSE score of the validation set reduced to 0.0658 by just increasing the hidden units size to 400.

Using the optimal hidden units, the learning rate was then optimised and the results can be seen in Figure 8. The optimal learning rate was found to be 0.001 with the RMSE of the model increasing slightly to 0.0660.

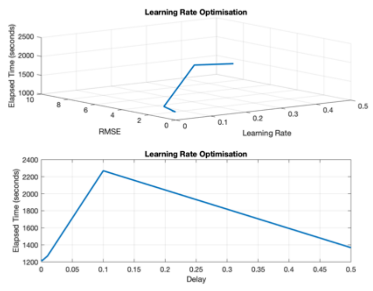
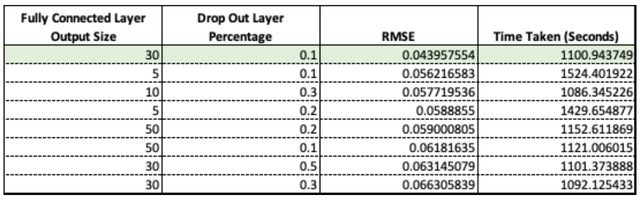


Figure 8 LSTM Grid Search: Learning Rate Optimisation

The dropout layer accompanied the fully connected layer as this reduces overfitting the LSTM model [15]. Adding extra depth to the model increased the calculation time of the neural network model considerably, initially the model took 60.40 seconds to compute, however looking at the optimal model the time taken was 1100.94 seconds. The RMSE of the validation dataset did fall to 0.0440, however no additional layers were added on top to the final model, as it was very computationally expensive.



Fully Connected Layer Output Size	Drop Out Layer Percentage	RMSE	Time Taken (Seconds)
30	0.1	0.043957554	1100.943749
5	0.1	0.056216583	1524.401922
10	0.3	0.057719536	1086.345226
5	0.2	0.0588855	1429.654877
50	0.2	0.059000805	1152.611869
50	0.1	0.06181635	1121.006015
30	0.5	0.063145079	1101.373888
30	0.3	0.066305839	1092.125433

Figure 7 LSTM Grid Search: Subset of Fully Connected Layer and Drop Out Layer Optimisation

Although using the hyperparameters from the Grid Search improved the accuracy of the model, the hyperparameters from Bayesian Optimisation, improved the LSTM network further. The optimal hyperparameters were 380 hidden units, 0.0019396 learning rate, 49 output size for Fully Connected Layer and 0.11953 for the drop out layer percentage. Bayesian optimisation was able to find hyperparameters to a higher degree of accuracy.

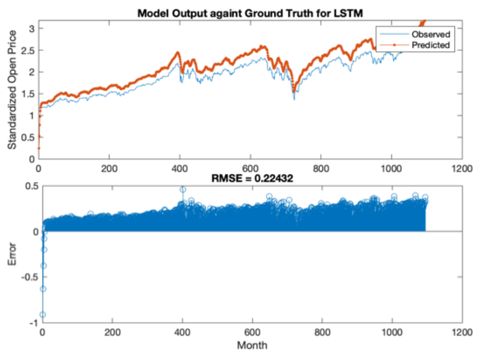


Figure 9 LSTM Model Output against Ground Truth

As seen from Figure 9 the LSTM network was able to fairly closely predict the test data. The RMSE score is shown to be 0.22432 .The RMSE on the test data is higher than that of the validation dataset which could mean that the LSTM network had overfitted on the training dataset. Looking at the error graph of how the LSTM network fits the test data, the errors were higher at the most recent dates compared to the earlier times. Surprisingly the LSTM network was able to model the sudden drops and increases in the S&P 500 stock index, but not to the same magnitude as the actual Opening price.

NARX

The optimal hyperparameters for the NARX model using a grid search were using a Bayesian Regularization training function, 75 hidden layers, input delay 20 and feedback delay of 2.

A base NARX model was created where the RMSE score on the validation dataset was 0.0931 and the time taken was 4.4166 seconds. The first hyperparameter to be optimised was the training function where the Bayesian Regularisation training function showed the best RMSE score of 0.0187 and time taken was 2.6989 seconds.

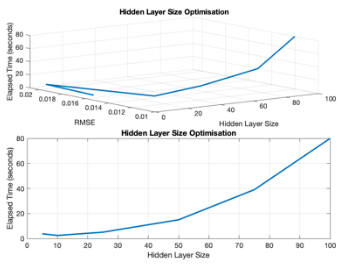


Figure 10 Hidden Layer Size Optimisation

The hidden layer size was the next hyperparameter optimised, where 75 was the optimal hidden layer size, which can be seen in Figure 10. The RMSE score was 0.0117 and time taken was 39.0727 seconds.

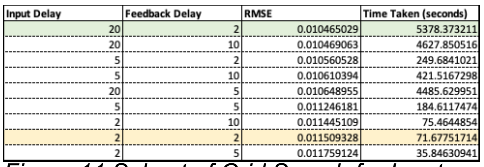
A grid search approach was taken to optimise the input delay and feedback delay as seen in Figure 11, where it was found that the input delay of 20 and feedback delay of 2 had the lower RMSE of 0.010465 however the time taken to optimise this was 5378.37 seconds.

Figure 12 shows the performance of the optimised NARX model against the ground truth, the test set. The Grid search optimal hyperparameters had a lower RMSE of 0.0569 compared to the hyperparameters chosen through Bayesian Optimisation which had a RMSE score of 0.9993.

When comparing the LSTM hyperparameter optimisation using grid search and Bayesian optimisation, Bayesian optimisation produced a higher accuracy (lower RMSE). This ties in with earlier where we stated the Bayesian optimisation has been shown to outperform other optimisation techniques [21]. Bayesian optimisation reduced the number of models that were run to find the optimal LSTM model. 100 iterations were run, but the model converged after 30 which can be seen in Figure 13. The grid search for the LSTM was exhaustive and still did not produce an accurate enough prediction for the LSTM model. However, when looking at the NARX model it was the Grid Search that produced a higher accuracy model compared to the Bayesian optimisation.

The initial hypothesis that LSTM would be the better neural network model in predicting the S&P 500 Opening price has been proven incorrect. The NARX model produced the better prediction as seen in Figure 12, with an RMSE score of 0.0313. This contradicts what has been shown previously [23], where LSTM had a better performance in multivariate time series compared to the NARX model. The second hypothesis of the NARX model being computationally quicker has also been proven incorrect where the time taken for the final NARX model was 5018 seconds and the LSTM model was 1347 seconds. The reason for why the NARX model took longer to compute is because the training function used for the final model was Bayesian Regularization. If Levenberg-Marquardt training function had been used instead, it would have been able to compute quicker [25].

The RMSE score in the final LSTM model for the test set was 0.22432, which was higher than the validation set in the optimal model with an RMSE score of 0.0019396. This shows signs of overfitting as the test set RMSE is higher than the validation set. In addition the LSTM model overpredicted the opening price, which can be seen in Figure 9. At nearly every data point, the model predicted higher than the actual price. Furthermore, the error gradually increased over time, which shows that the long term prediction of time series models are very challenging. This trend can also be seen in other studies [24].



Input Delay	Feedback Delay	RMSE	Time Taken (seconds)
20	2	0.010465029	5378.373211
20	10	0.010469063	4627.850516
5	2	0.010560528	249.6841021
5	10	0.010610394	421.5167298
20	5	0.010648955	4485.629951
5	5	0.011246181	184.6117474
2	10	0.011445109	75.4644854
2	2	0.011509328	71.67751714
2	5	0.011759124	35.84630941

Figure 11 Subset of Grid Search for Input delay and Feedback Delay

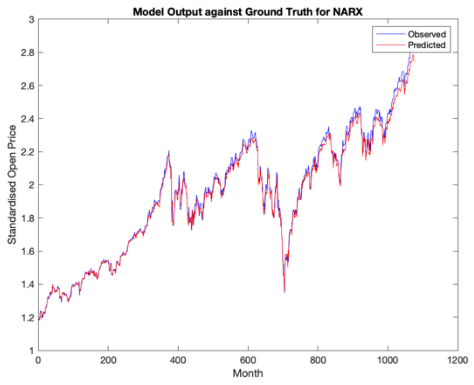


Figure 12 Model Output against Ground Truth NARX

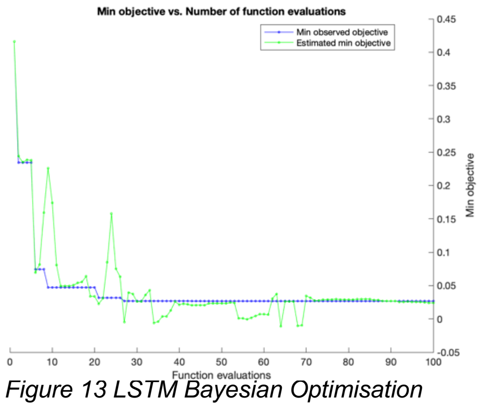


Figure 13 LSTM Bayesian Optimisation

Conclusion and Future Work

This paper critically evaluated both the LSTM and NARX models and shows how well the optimised models can predict the S&P 500 index. The NARX model accurately predicted the Opening price. Although, after 'optimising' both models, it did not seem to accurately predict the S&P 500 index. Even a very small error from the models can result in huge losses when trading the index; hence more research needs to be initiated and implemented in order to reduce the error produced. To the eye, these models may look perfect however at longer time periods the error rate increased. In addition both models predicted peaks and troughs a day or two late as well as not predicting it to the correct magnitude. In industry, this will create large losses for institutions or individual investors.

To obtain precise results, other factors could have been included into both the LSTM and NARX models, such as unemployment rates and interest rates, which all affect the stock price movements, and these could have been included to obtain a more accurate prediction.

Unfortunately, this was out of scope for this paper as it was computationally expensive. In addition, using forward chaining as a cross validation technique could have produced more accurate results, in order to split the data into different sections such as bear and bull markets. Further, for the LSTM network, a deeper LSTM network could have been included, by adding more layers into the network to see if a better prediction would have been obtained. However, this was out of scope for this paper due to the additional computational resources required to complete this.

Each stock index or stock moves differently to one another, so in the future perhaps both these models could accurately predict another stock index such as the FTSE 100. Additionally, the forex market is affected by additional factors on top of what affects the stock indices, hence we could see if these optimised models can predict the forex market such as GBP/USD or GBP/EUR. Other factors should be included to provide a more accurate prediction of different stock indices such as sentiment of news articles or headlines.

References

[1] Us.spindices.com. (2020). *S&P 500® - S&P Dow Jones Indices*. [online] Available at: https://us.spindices.com/indices/equity/sp-500.

[2] PyPI. (2020). *yfinance*. [online] Available at: https://pypi.org/project/yfinance/.

[3] Programcreek.com. (2020). *pandas_datareader.data.DataReader Python Example*. [online] Available at: https://www.programcreek.com/python/example/92134/pandas_datareader.data.DataReader.

[4] Ayodele A., A., Charles K., A., Marion O., A. and Sunday O., O. (2012). *Stock Price Prediction using Neural Network with Hybridized Market Indicators*. [online] Eprints.covenantuniversity.edu.ng. Available at: http://eprints.covenantuniversity.edu.ng/4112/1/Emerging_Trend.pdf.

[5] Ahangar, R., Yahyazadehfar, M. and Pournaghshband, H. (2010). *The Comparison of Methods Artificial Neural Network with Linear Regression Using Specific Variables for Prediction Stock Price in Tehran Stock Exchange*. [online] Arxiv.org. Available at: https://arxiv.org/ftp/arxiv/papers/1003/1003.1457.pdf.

[6] Yu, Lean & Huang, Wei & Lai, Kin Keung & NAKAMORI, YOSHITERU & Wang, Shouyang. (2007). Neural Networks in Finance and Economics Forecasting. International Journal of Information Technology & Decision Making (IJITDM). 06. 113-140. 10.1142/S021962200700237X.

[7] Olah, C. (2015). *Understanding LSTM Networks -- colah's blog*. [online] Colah.github.io. Available at: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[8] Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), pp.1735-1780.

[9] Brownlee, J. (n.d.). *Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/.

[10] Gudikandula, P. (2019). *Recurrent Neural Networks and LSTM explained*. [online] Medium. Available at: https://medium.com/@purnasaigudikandula/recurrent-neural-networks-and-lstm-explained-7f51c7f6bbb9.

[11] Shahbazi, Nima & Memarzadeh, Masoud & Gryz, Jarek. (2016). Forex Market Prediction Using NARX Neural Network with Bagging. MATEC Web of Conferences. 68. 19001. 10.1051/matecconf/20166819001.

[12] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15, pp.1929 - 1958.

[13] Uk.mathworks.com. 2020. *Design Time Series NARX Feedback Neural Networks- MATLAB & Simulink- Mathworks United Kingdom*. [online] Available at: <https://uk.mathworks.com/help/deeplearning/ug/design-time-series-narx-feedback-neural-networks.html>.

[14] Diaconescu, Eugen. (2008). The use of NARX neural networks to predict chaotic time series. WSEAS Transactions on Computer Research. 3.

[15] Brownlee, J., 2020. *A Gentle Introduction To Dropout For Regularizing Deep Neural Networks*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/>.

[16] Goodfellow, I., Bengio, Y. and Courville, A., 2016. Deep Learning (Adaptive Computation And Machine Learning Series). MIT Press.

[17] Bushaev, V., 2018. Adam—Latest Trends In Deep Learning Optimization.. [online] Medium. Available at: <https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c>.

[18] Tsungnan Lin, B. G. Horne, P. Tino and C. L. Giles, "Learning long-term dependencies in NARX recurrent neural networks," in IEEE Transactions on Neural Networks, vol. 7, no. 6, pp. 1329-1338, Nov. 1996.

[19] A. A. Ferreira, T. B. Ludermir and R. R. B. de Aquino, "Comparing recurrent networks for time-series forecasting," in The 2012 International Joint Conference on Neural Networks (IJCNN), Brisbane, 2012.

[20] Brownlee, J., 2020. Crash Course In Recurrent Neural Networks For Deep Learning. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/crash-course-recurrent-neural-networks-deep-learning/>.

[21] Snoek, J., Larochelle, H. and Adams, R., 2012. *Practical Bayesian Optimization Of Machine Learning Algorithms*. [online] Papers.nips.cc. Available at: <https://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf>.

[22] Montavon, G., Orr, G. and Müller, K., 2012. *Neural Networks: Tricks Of The Trade*. 2nd ed. Berlin: Springer, pp.53-67.

[23] G. Abbas, M. Nawaz and F. Kamran, "Performance Comparison of NARX & RNN-LSTM Neural Networks for LiFePO4 Battery State of Charge Estimation," 2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST), Islamabad, Pakistan, 2019, pp. 463-468.

[24] Kamal, Imam & Bae, Hyerim & Sunghyun, Sim & Yun, Heesung. (2020). DERN: Deep Ensemble Learning Model for Short- and Long-Term Prediction of Baltic Dry Index. Applied Sciences. 10. 1504. 10.3390/app10041504.

[25] Kayri, Murat. (2016). Predictive Abilities of Bayesian Regularization and Levenberg–Marquardt Algorithms in Artificial Neural Networks: A Comparative Empirical Study on Social Data. Mathematical and Computational Applications. 21. 1-11. 10.3390/mca21020020.

Appendix 1 – Glossary

Adj. Close Price – amended stock's closing price to accurately reflect the value after accounting for any corporate actions
Back propagation – algorithm used in training feedforward neural networks for supervised learning.
Bayesian Optimisation - algorithm attempts to minimize a scalar objective function *f*(*x*) for *x* in a bounded domain. The function can be deterministic or stochastic, meaning it can return different results when evaluated at the same point *x*. *x* can be continuous reals, integers, or categorical, meaning a discrete set of names.
Close Price – Closing price of stock for the day
Dropout Layer – randomly sets input elements to zero with a given probability
Forward Chaining (rolling origin) - is a method of cross validation for time series.
Gradient Descent – an optimisation algorithm used to minimise a function by moving in the direction of the steepest descent. This is used to update the parameters of a neural network
Grid Search is the process of using the data to find the optimal hyperparameters for a given model.
Early Stopping – method to avoid overfitting when training a neural network
Epoch – number of passes through the entire training dataset
Fully Connected Layer – multiplies the input by a weight matrix and then adds a bias vector
Hidden Units – number of LSTM units in a LSTM cell at every step of the network
High Price – the highest price of the security in that day
Hyperparameter Tuning – finding the optimal hyperparameters for a given model
Learning Rate – tuning parameter that determines the step size at each iteration whilst moving towards a minimum loss function
Low price – the lowest price of the security in that day
LSTM – Long Short-Term Memory Network. The LSTM network trains using Backpropagation Through Time which overcomes the vanishing gradient problem
LSTM Layer – learns long term dependencies between time steps in time series and sequence data
NARX - Nonlinear Autoregressive Network with Exogenous Inputs. The NARX model is a recurrent dynamic network, with feedback connections enclosing several layers of the network
Open Price – price at which the security first trades when the market opens for the day
Regression Layer - computes the half-mean-squared-error loss for regression problems
RMSE - Root Mean Squared Error is the standard deviation of the difference between the true value and the prediction (prediction error)
RNN – Recurrent Neural Network
Sequence Input Layer – inputs sequence data to a network
Volume – amount of units of the stock traded in a day

Appendix 2 – Implementation Details

1. **inputData.m** loads the train, validation and test data for the LSTM model.
2. **inputNarx.m** loads the train and test data for the NARX model.
3. **LSTM_BayesOpt.m** – creates the function for the LSTM model to perform Bayesian optimisation
4. **NARX_BayesOpt.m** – creates the function for the NARX model to perform Bayesian Optimisation
5. **LSTM_NARX.m** – Includes base model, optimisation steps and final model for LSTM and NARX
6. **FinalBestModels.m** – Contains the best models for both LSTM and NARX with test data
7. **BEST_LSTM_Model.mat** – final optimised LSTM model
8. **BEST_NARX_Model.mat –** final optimised NARX model
9. **TestData.xlsx –** contains the test data
10. **SP500.csv** – original dataset obtained through yfinance package in Python
11. **SP500_standardise.csv** – original dataset standardised
12. **Neural Computing Coursework.ipynb** – data collection and pre-processing

Open functions 1 – 4 in order to run script 5. Script 6 allows to test the final optimised models using 7 & 8 which contains the final optimised models for LSTM and NARX.

12 was used to obtain the data using the yfinance package and pre-processing was done in python such as standardising the data.