International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-8, Issue-5, May 2022
ISSN: 2395-3470
www.ijseas.com

Comparison Study: On Using Machine and Deep Learning Approaches To Compare Performance and Prediction Error Rates for Stock Price

Mohsen A. Hassan[1] , Aliaa AA Youssif[2] , Osama Imam[3] , Amr S. Ghoneim[4],

[1] Researcher –PHD-, Faculty of Computers & Artificial Intelligence, Helwan University (HU), Cairo, Egypt.
[2] Professor of Computer Science, Arab Academy for Science, Technology and maritime transportation (AASTMT), Cairo, Egypt.
[3] Professor of information system, Faculty of Computers & Artificial Intelligence, Helwan University (HU), Cairo, Egypt.
[4] Assistant Professor of Computer Science, Faculty of Computers & Artificial Intelligence, Helwan University (HU), Cairo, Egypt.

Abstract

Wealth management advisory is an essential aspect of a bank's services nowadays. Wealth management provides financial planning and investment strategy to sustain and grow one's wealth. Investing in stocks is a crucial component of Wealth management since it provides analysis and recommendations on where to invest. Anticipating the future value of a company's stock or other financial instruments traded on an exchange is known as the stock market prediction. Stock prediction research has gained prominence as Machine and Deep learning become more prevalent. Investors may reap significant profits if they can accurately predict closing stock prices. Machine learning algorithms can evaluate historical stock movements and anticipate closing prices with near-perfect accuracy. In this research, we use a variety of machine and deep learning approaches to compare performance and prediction error rates. From our observation, it is evident that Deep learning-based models outperformed their Machine learning counterparts, yielding better Root Mean Square Errors on four distinct datasets. The best performing model is the LSTM-based model, yielding the lowest RMSEs across the four datasets.

Keywords: **Wealth Management advice, Gradient Boosting, Long Short-Term Memory (LSTM), Stock Price Prediction.**

1. Introduction

Individuals and families ranging from affluent to high-net-worth (HNW) and ultra-high-net-worth (UHNW) individuals and families are served by wealth management (WM) or wealth management advice (WMA) [1, 2, 3, 4, 5]. It is a discipline that entails planning and structuring assets to aid in the growth, preservation, and protection of wealth while passing it on to the family in a tax-efficient manner and following their intentions. Tax planning, wealth protection, estate planning, succession planning, and family governance are all part of wealth management. One of the ways wealth management advisors recommend growing one's wealth is through investing in the stock market. The stock market is an essential part of a country's economic development [6, 7]. A corporation can raise a significant amount of money through an Initial Public Offering to grow its operations. It's an opportunity for investors to purchase a brand-new stock and become either a stockholder who benefits from the firm's shareholder bonus program or a stock trader who trades stock on the stock market. The stock trader would make huge gains if he correctly predicted stock price patterns. The stock market, on the other hand, is volatile. Daily news events such as evolving political conditions, company performance, and other unforeseen events have an immediate favorable or unfavorable impact on stock values [8, 9, 10]. As a result, accurately predicting stock prices and their directions (growth, drop) is impossible; instead, investors must foresee forthcoming short-term trends. Before buying stock, an investor assesses a company's performance. To prevent buying overvalued or high-risk stocks, the evaluation comprises examining a company's quarterly earnings report and paying attention to the news. However, in recent years, both the rate of publication and the number of daily news providers have increased dramatically [11, 12, 13], exceeding investors' abilities to examine massive amounts of data. As a result, an automated decision support system is mandatory as it analyses and forecasts

future stock developments. Stock market prediction [14]is undoubtedly one of the most challenging issues for traders and researchers. The opportunities of high profit offered by the stock market have always attracted a large number of investors. Researchers consider stock market prediction as a challenging task due to the difficulty in capturing the nonlinear and non-stationary variation in data. The applications of the machine and deep learning [15, 16] techniques for stock market prediction are a well-established area. With the rising popularity of Machine and Deep learning [17, 18], it is essential to harness the state-of-the-art models for stock

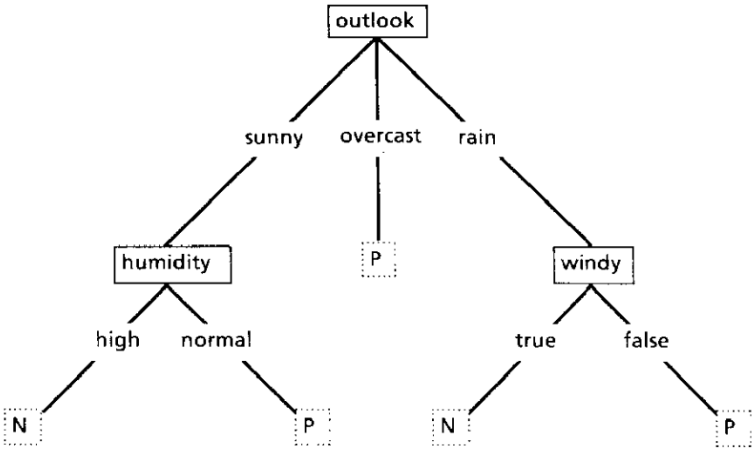

Figure 1: Simple Decision Tree. Source: Figure 2 of [21]

market prediction. In this paper, we investigate how modern machine and deep learning techniques can be utilized for stock market prediction while testing on four datasets. After stating the motivations for wealth management advice, and the utilization of Machine and Deep learning to employ them in the process, the remaining part of the paper is organized as follows. Different Machine/Deep Learning approaches for stock market prediction are reviewed in Section 2. Section 3, shows the employed dataset, and evaluation metrics. The results and experiments for both Machine and Deep Learning architectures are presented in Section 4. The conclusion and further work are found in Section 5.

2. Proposed Methodology

In the forecasting community, machine learning models have attracted attention and established themselves as strong competitors to classical statistical models [19, 20]. These models, also known as black-box or data-driven models, These black box or data driven models are examples of nonparametric non linear models which learn the stochastic dependency between the past and the future using only historical data[22] , k-nearest neighbour regression [23], and Gradient Boosting [24]. Moreover, the empirical correctness of numerous machine learning models has been investigated in a variety of diverse data situations outperforming conventional models.

2.1. Machine Learning

2.1.1 KNN

The k-nearest neighbors (KNN) algorithm [23, 25] is a supervised machine learning algorithm that can tackle classification and regression problems. It’s simple to set up, but it has the major disadvantage of being substantially slower as the amount of data in use grows. KNN works by calculating the distances between a query and all of the instances in the data, picking the K closest examples to the query, and then voting for the most frequent label (in the case of classification) or averaging the labels (in the case of regression). Because the stock market prediction problem Neils heavily towards regression, we chose KNN as a regressor for our purposes.

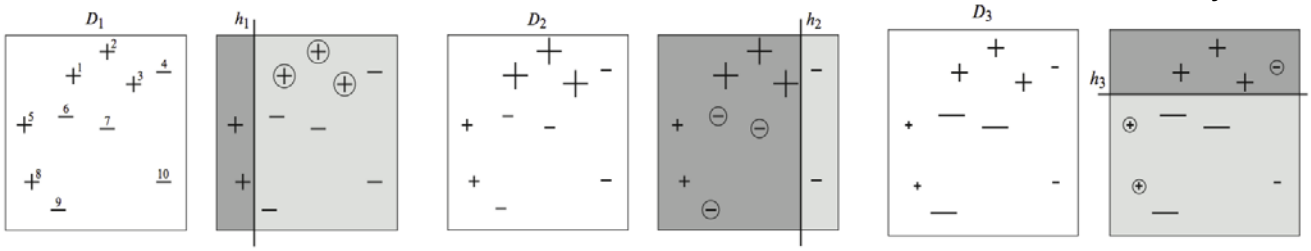

Figure 2: Overview of Boosting Process. Source: Figure 1.1 of [29]

2.1.2. Decision Tree

For classification and regression, Decision Trees (DTs) [21, 26] are a nonparametric supervised learning method. The goal is to learn simple decision rules from data attributes to develop a model that predicts the value of a target variable. A tree is an approximation of a piecewise constant. A decision tree can be used to depict decisions and decision-making visually and explicitly. A decision tree is a flowchart-like structure in which each internal node represents a feature test, each leaf node represents a class label, and branches represent feature combinations leading to those class labels. A simple Decision tree structure is shown in Figure 1.

2.1.3. Random Forest Regressor

Random forest [27, 28] is a flexible, easy-to-use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of its simplicity and diversity. Random forest is a supervised learning algorithm. The forest is an ensemble of decision trees, usually trained with the "bagging" method. The general idea of the bagging method is that a combination of learning models increases the overall result. Random forests are also very hard to beat performance-wise compared to other machine learning algorithms. Random forests were the third-best result across the four datasets.

2.1.4. Light Gradient Boosting Machine

Light Gradient Boosted Machine (LightGBM) [30, 31] is an open-source gradient boosting implementation that aims to be more efficient and effective than other approaches. LightGBM is based on Gradient-based One-Side Sampling (GOSS), a variation of the gradient boosting approach that focuses emphasis on training samples that result in a greater gradient, speeding up learning and lowering the method's computational complexity. When combined with Exclusive Feature Bundling (EFB), a method for bundling sparse (mainly zero) mutually exclusive features, such as one-hot encoded categorical variable inputs. As a result, it's a form of feature selection that's done automatically.

2.1.5. Gradient Boosting Regressor

Gradient boosting [32, 33] is a machine learning technique for regression, classification, and other problems that generates a prediction model from an ensemble of weak prediction models, most commonly decision trees. When a decision tree is a poor learner, the resulting method is known as gradient boosted trees, and it outperforms random forest in most cases. It constructs the model in the same stage-by-stage manner as other boosting approaches, but it broadens the scope by allowing optimization of any differentiable loss function. Gradient boosting combines the estimates of a group of simpler, weaker models to attempt to properly forecast a target variable.

2.1.6. AdaBoost Regressor

An AdaBoost regressor [34, 35] is a meta-estimator that starts by fitting a regressor on the original dataset, then fits further copies of the regressor on the same dataset, but with the weights of instances changed based on the current prediction's error. As a result, future regressors concentrate on the most challenging cases. AdaBoost is a machine learning algorithm that may be used to improve the performance of any other machine learning technique. It works well with students who are struggling. On a classification task, these are models that reach accuracy just above random chance. Decision trees with one level are the most suitable and hence the most commonly used algorithm with AdaBoost.

2.1.6. AdaBoost Regressor

An extra-trees regressor [36, 37]is a meta estimator that employs averaging to increase predictive accuracy and control over-fitting by fitting many randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset. The Extra Trees approach uses the training dataset to generate a huge number of unpruned decision trees. In the case of regression, predictions are formed by averaging the prediction of the decision trees, whereas, in the case of classification, majority voting is used. The Extra Trees technique, like random forests, will sample characteristics at each split point of a decision tree at random. Unlike random forest, which selects an optimal split point using a greedy method.

2.2. Deep Learning

Deep learning algorithms have recently outperformed standard models in many machine learning problems [38]. Deep neural networks have been effectively applied to time series forecasting problems, which is a critical area in data mining [39]. Because of their ability to automatically understand the temporal connections found in time series, and also the property of parameter sharing that provides borrow statistical strength from each other. Therefore, they have shown to be an effective solution. For all Deep learning algorithms, we train on 70% of each dataset, while 15% is for validation, and 15% for testing. For prediction, the input is a sample containing the last 50 days of closing prices and the output in the prediction of the price on day 51. We trained for 100 epochs with a batch size of 64 and we chose Adam as our optimizer.

2.2.1. Long Short-Term Memory (LSTM)

In the realm of deep learning, Long Short Term Memory (LSTM) [40, 41]is an artificial recurrent neural network (RNN) architecture. Short-term memory is a feature of such networks, and the hypothesis to test here is that this feature can provide improvements in terms of results when compared to other classic Machine Learning methodologies. LSTM has feedback connections, unlike traditional feed-forward neural networks. A cell, an input gate, an output gate, and a forget gate make up a typical LSTM unit. The gate in the cell regulates the flow of information in the cell, and the cell remembers values across random time intervals. Because LSTM is best suited for time series analysis, it is preferred above other Neural networks such as Recurrent Neural networks (RNN). Each LSTM contains three types of gates that govern the state of each cell: Forget Gate returns a value between 0 and 1, with 1 indicating ”totally keep this” and 0 indicating ”absolutely ignore this.” Memory Gate determines which fresh data must be saved in the cell. First, the ”input door layer,” a sigmoid layer, determines which values will be changed. A real layer then generates a vector of new candidate values that can be added to the state. The Output Gate determines how much each cell will generate. The resulted value will be determined by the cell status, as well as filtered and freshly added data.

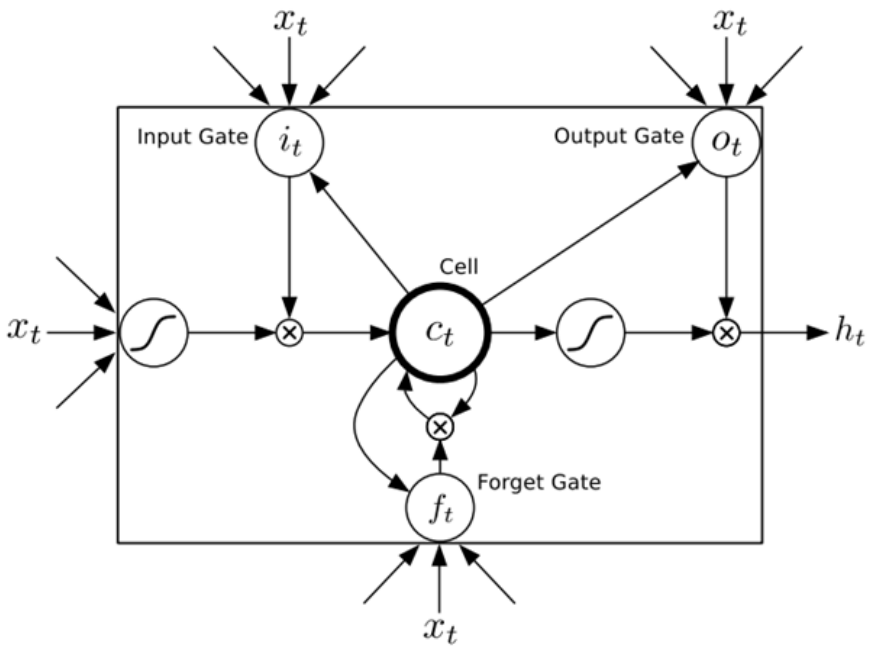

Figure 3: Long Short-term Memory Cell.

2.2.2. Bi-LSTM

A bidirectional LSTM, often known as a bi-LSTM [40], is a sequence processing model that consists of two LSTMs, one of which takes input in one way and the other in the other. The amount of information available to the network is efficiently increased by Bi-LSTMs. Bidirectional LSTMs are a type of LSTM that can be used to increase model performance in sequence classification issues. Bidirectional LSTMs train two instead of one LSTM on the input sequence in instances where all time steps of the input sequence are available. The first is based on the original input sequence, and the second is based on a reversed replica of the original input sequence. This can give the network more context and help it learn the problem faster and more thoroughly.

2.2.3. Gated Recurrent Units (GRUs)

In recurrent neural networks, gated recurrent units (GRUs) [42] are a gating mechanism. The GRU is similar to a long short-term memory (LSTM) with a forget gate, but it lacks an output gate, hence it has fewer parameters. On some smaller and less frequent datasets, GRUs have been found to perform better. The goal of GRE is to tackle the vanishing gradient problem that a normal recurrent neural network has. The GRUs abandoned the cell state in favor of using the hidden state to transfer data. It also only has two gates, one for reset and one for the update. The update gate functions similarly to an LSTM forget and input gate.

2.2.4. Bi-Gated Recurrent Units (GRUs)

A Bidirectional GRU, or BiGRU, is a sequence processing model that consists of two GRUs, one taking the input in a forward direction, and the other in a backward direction. It is a bidirectional recurrent neural network with only the input and forgets gates. Bi-GRU functions similarly to a BiLSTM, giving the network more context and helping it learn the problem faster and more thoroughly.

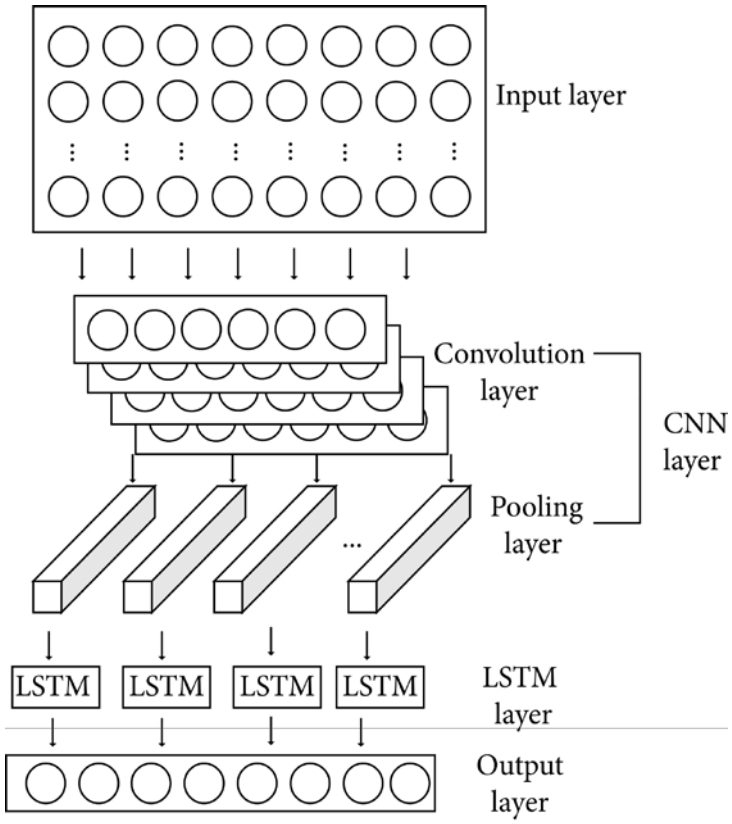

Figure 4: CNN-LSTM Architecture

2.2.5. CNN-LSTM

The Convolutional Long Short-Term Memory Network, or CNN LSTM [43] for short, is a type of LSTM architecture intended primarily for sequence prediction issues including spatial inputs. Convolutional Neural Network (CNN) [44] layers for feature extraction on input data are paired with LSTMs to facilitate sequence prediction in the CNN LSTM architecture. It can be used to forecast time series with great success. CNN's local perception and weight sharing can considerably minimize the number of parameters, enhancing model learning efficiency. In principle, this is a significant improvement over employing a conventional LSTM, however, in our tests, it performed worse than a regular LSTM. CNN-LSTM structure is illustrated in Figure 4.

2.2.6. Attention Models

In natural language processing, the attention mechanism [46] outperformed the encoder decoder-based neural machine translation system (NLP). This approach, or adaptations of it, was later applied in other applications such as computer vision, speech processing, and so on. The LSTM encoder processes the full input sentence and encodes it into a context vector, which is the LSTM/final RNN's hidden state. This should be an accurate summary of the input sentence. All of the encoder's intermediate states are ignored, and the final state ID is expected to be the decoder's first hidden state. The LSTM or RNN units in the decoder output the words in a sentence one after the other. It passes the summary (context vector) to the decoder which translates the input sentence by just seeing it. The Bi-LSTM framework now includes two attention methods. In traditional LSTM, the forget gate is replaced by the attention gate. It only applies to the previous cell state and has no bearing on the present input. As a result, the attention gate can minimize the number of training parameters significantly.

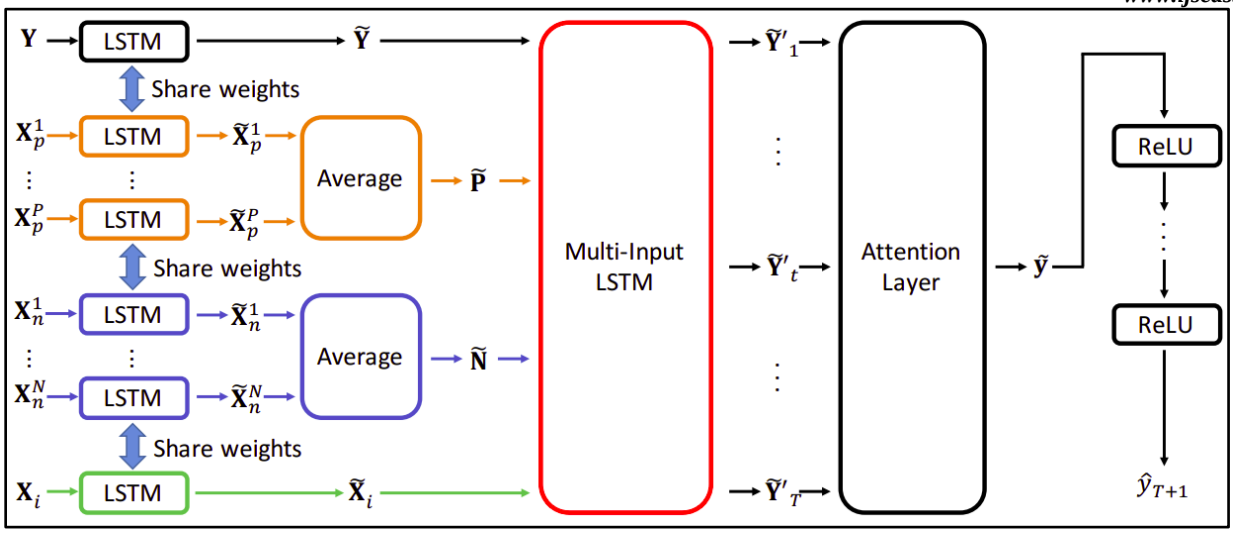

Figure 5: Illustration of Attention-LSTM Model [45]

Furthermore, the attention weighting method is applied to Bi-LSTM output, allowing it to investigate the most important data. This takes a similar approach to the attention-LSTM variation but adds BiLSTMs to the mix. We followed a similar approach [45] of attention-LSTMs, combining an attention layer with a GRU and a BiGRU to get the best of both worlds while covering as many variants of the GRU as possible. The employed model is illustrated in Figure 5.

3. Experimental Work

We chose PyCaret to implement the machine learning algorithms. PyCaret is a Python machine learning package that automates machine learning operations and is open-source. It's an end-to-end machine learning and model management solution that exponentially speeds up the trial cycle and increases productivity. PyCaret is a low-code alternative to the major open-source machine learning tools, allowing you to replace hundreds of lines of code with just a few words. Experiments become progressively faster and more efficient as a result of this. For all machine learning algorithms, we train on 70% of each dataset, while 15% is for validation, and 15% for testing. For prediction, the input is a sample containing the last 50 days of closing prices and the output in the prediction of the price on day 51.

3.1. The Constructed Dataset

For our datasets, we have collected four distinct datasets from four different destinations. The first is from the Commercial International Bank- Egypt, covering stocks from the second of February 2012 to the eleventh of February 2021. The second is from Hadioslb, covering stocks from the first of January 2014 to the fourth of February 2021. The third is from Orascom Hotels and Development, covering stocks from the first of January 2015 to the fourth of February 2021. The final dataset is from Palm-Hills Development, covering stocks from the first of January 2012 to the fourth of February 2021. Each dataset contains the date of the reading, the opening price, the high value of the stock, the low value of the stock, and the volume. These datasets did not contain a closing price, so we had to calculate the closing price for each record in the datasets.

3.2. Evaluation Metric

The root-mean-square error (RMSE) [47, 48] is the evaluation metric we utilize. RMSE is a commonly used measure of the variations between values predicted by a model or estimator and the values observed. The square root of the second sample moment of the discrepancies between anticipated and observed values, or the quadratic mean of these differences, is represented by the RMSD.

Figure 6: Data Separation

The magnitudes of the mistakes in predictions for numerous data points are combined into a single measure of predictive power called RMSD. Because it is scale-dependent, RMSD is used to evaluate predicting errors of different models for a specific dataset rather than between datasets. The square root of the average of squared mistakes is the RMSD. Each error's effect on RMSD is proportional to the squared error's size. Consequently, larger errors have a disproportionately large impact on RMSD. As a result, RMSD is susceptible to outliers.

Table 1: Sample Data

Price	***Open***	***High***	***Low***	***Close***
19.01	21.07	20.13	21.33	20.18
21.12	23.17	23.17	20.93	23.17
34.33	34.51	34.79	34.01	34.51
36.18	35.96	36.64	35.86	35.96
38.06	37.69	38.41	37.62	37.69

4. Results and Experiments

In this section, we illustrate the models' predictions and show their forecasts compared to the ground truth values. From our observation, it is evident that Deep learning-based models performed better than their Machine learning counterparts across the thirteen experiments. While the differences are minor, it still proves that Deep learning models are better at catching and learning specific features that can give the edge for their prediction. In addition, LSTM-based models proved to be more accurate than the rest of the models, yielding the least RMSE across all datasets, followed closely by GRU-based models. While Machine learning algorithms were faster, they did not yield RMSE scores close to their Deep learning counterparts. The best performing Machine learning-based algorithm was the Gradient Boosting Regressor, followed closely by the Extra Trees Regressor, the Random Forest Regressor, and the Decision Tree. Illustrations of the best performing models and the RMSEs of the other tested models are shown below.

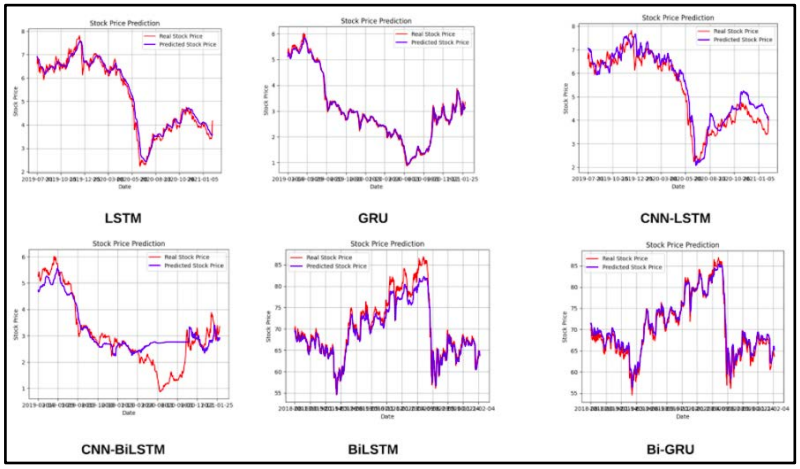

Figure 7: Proposed DL Models

To summarize, for the COMI dataset, the best-performing machine learning model is the Gradient Boosting Regressor with an RMSE of 0.7442 while the best performing deep learning model is the LSTM with an RMSE of 0.0259. For the IRON dataset, the best-performing machine learning model is the Extra Trees Regressor with an RMSE of 0.0451 while the best performing deep learning model is the LSTM with an RMSE of 0.0438. For the ORHD dataset, the best-performing machine learning model is the Gradient Boosting Regressor with an RMSE of 0.1134 while the best performing deep learning model is the Bi-LSTM with an RMSE of 0.0054. For the PHDC dataset, the best performing machine learning model is the Gradient Boosting Regressor with an RMSE of 0.0479 while the best performing deep learning model is the Bi-LSTM with an RMSE of 0.01558.

Table 2: ML Algorithms Results

Algorithm	***COMI***	***IRON***	***ORHD***	***PHDC***
KNN	19.7788	2.766800-e	3.1308	0.9348
Decision Tree	0.9428	0.047	0.1211	0.0623
XGBOOST	0.8338	0.0609	0.1396	0.0558
Random Forest Regressor	0.7743	0.0501	0.1146	0.0489
Light Gradient Boosting Machine	0.8472	0.1071	0.2553	0.0645
Gradient Boosting Regressor	0.7442	0.047	0.1134	0.0479
AdaBoost Regressor	1.0758	0.1547	0.3064	0.0709
Extra Trees Regressor	0.7913	0.0451	0.1779	0.0491

Table 3: DL Algorithms Results

Algorithm	*COMI*	*IRON*	*ORHD*	*PHDC*
LSTM (50 Hidden Units)	0.0355	0.0904	0.0083	0.0276
LSTM (256 Hidden Units)	0.0259	0.0438	0.0067	0.0262
Bi-LSTM (50 Hidden Units)	0.04332	0.0826	0.0826	0.0178
Bi-LSTM (256 Hidden Units)	0.0315	0.0706	0.0706	0.01558
GRU(50 Hidden Units)	0.02601	0.0631	0.0094	0.0163
Bi-GRU(50 Hidden Units)	0.0290	0.0792	0.0058	0.0229
CNN-LSTM	0.0457	0.4937	0.0062	0.114
CNN-Bi-LSTM	0.04775	0.404	0.00755	0.094
Attention-LSTM	0.0659	0.0615	0.0172	0.0271
Attention-Bi-LSTM	0.1049	0.06249	0.0185	0.038
Attention-GRU	0.0906	0.05707	0.0201	0.0236
Attention-Bi-GRU	0.0651	0.1269	0.0190	0.0257

5. Conclusion and Future Work

In this paper, we compared 14 Machine and Deep Learning architectures on four datasets. Results indicate that Deep Learning architectures outperform classical Machine Learning algorithms by a noticeable margin. In our future work, we aim to test on larger datasets while testing on state-of-the-art models. We also aim to perform sentiment analysis on stock market-related news which we believe could add to the robustness of the model's prediction.

References

[1] R. Ferdiana, S. Sulistyo, et al., The role of information technology usage on startup financial management and taxation, Procedia Computer Science 161 (2019) 1308–1315.
[2] T. Magoˇc, F. Modave, M. Ceberio, V. Kreinovich, Computational methods for investment portfolio: the use of fuzzy measures and constraint programming for risk management, in: Foundations of Computational Intelligence Volume 2, Springer, 2009, pp. 133–173.
[3] T. Cocca, et al., Potential and limitations of virtual advice in wealth management, Journal of Financial Transformation 44 (2016) 45–57.
[4] S. R. Das, D. N. Ostrov, A. Radhakrishnan, D. Srivastav, A new approach to goals-based wealth management, Available at SSRN 3117765 (2018).
[5] H. Mohamed, I-fintech and its value proposition for islamic asset and wealth management, in: Islamic

FinTech, Springer, 2021, pp. 249–266.
[6] Y. Kim, S. R. Jeong, I. Ghani, Mining to analyze news for stock market prediction, 2014.
[7] V. Ramalingam, A. A. Pandian, S. Dwivedi, J. Bhatt, Analysing news for stock market prediction, 2018.
[8] Y. Kim, M. Jeong, S. R. Jeong, Using big data opinion mining to predict rises and falls in the stock price index, in: Handbook of Research on Organizational Transformations through Big Data Analytics, IGI Global, 2015, pp. 30–42.
[9] A. Buche, D. M. B. Chandak, Stock market prediction using text opinion mining: A survey, 2016.
[10] Y. Kim, S. R. Jeong, Opinion-mining methodology for social media analytics, KSII Trans. Internet Inf. Syst. 9 (2015) 391–406.
[11] F. S. Alzazah, X. Cheng, Recent advances in stock market prediction using text mining: A survey, E-Business - Higher Education and Intelligence Applications (2020).
[12] A survey paper on stock price prediction using machine learning techniques, 2021.
[13] M.-A. Mittermayer, G. Knolmayer, 1 text mining systems for predicting market response to news : A survey, 2007.
[14] A. Sharma, D. Bhuriya, U. Singh, Survey of stock market prediction using machine learning approach, 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA) 2 (2017) 506–509.
[15] P. D. Yoo, M. H. Kim, T. Jan, Machine learning techniques and use of event information for stock market prediction: A survey and evaluation, in: International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), volume 2, IEEE, 2005, pp. 835–841.
[16] E. Chong, C. Han, F. C. Park, Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies, Expert Syst. Appl. 83 (2017) 187–205.
[17] L. dos Santos Pinheiro, M. Dras, Stock market prediction with deep learning: A character-based neural language model for event-based trading, in: ALTA, 2017.
[18] J. Shen, M. O. Shafiq, Short-term stock market price trend prediction using a comprehensive deep learning system, Journal of big Data 7 (2020) 1–33.
[19] I. Parmar, N. Agarwal, S. Saxena, R. Arora, S. Gupta, H. Dhiman, L. Chouhan, Stock market prediction using machine learning, in: 2018 first international conference on secure cyber computing and communication (ICSCCC), IEEE, 2018, pp. 574–576.
[20] R. Y. Sable, S. Goel, P. Chatterjee, Empirical study on stock market prediction using machine learning, 2019 International Conference on Advances in Computing, Communication and Control (ICAC3) (2019) 1–5.
[21] J. R. Quinlan, Induction of decision trees, Machine learning 1 (1986) 81–106.
[22] B. E. Boser, I. M. Guyon, V. N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, ????, pp. 144–152. [23] A. Ali, M. Hamraz, P. Kumam, D. M. Khan, U. Khalil, M. Sulaiman, Z. Khan, A k-nearest neighbours based ensemble via optimal model selection for regression, IEEE Access 8 (2020) 132095–132105.
[24] J. H. Friedman, Greedy function approximation: a gradient boosting machine, Annals of statistics (2001) 1189–1232.
[25] K. Alkhatib, H. Najadat, I. Hmeidi, M. K. A. Shatnawi, Stock price prediction using k-nearest neighbor (knn) algorithm, International Journal of Business, Humanities and Technology 3 (2013) 32–44. [26] M. Mir´o-Juli`a, G. Fiol-Roig, A. P. Isern-Dey`a, Decision trees in stock market analysis: Construction and validation, in: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Springer, 2010, pp. 185–194.
[27] S. Han, Stock prediction with random forests and long short-term memory (2019).
[28] L. Breiman, Random forests, Machine learning 45 (2001) 5–32.
[29] Y. Freund, R. Schapire, N. Abe, A short introduction to boosting, JournalJapanese Society For Artificial Intelligence 14 (1999) 1612.
[30] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, Lightgbm: A highly efficient gradient boosting decision tree, Advances in neural information processing systems 30 (2017).
[31] R. M. Nabi, S. Soran Ab M, H. Harron, A novel approach for stock price prediction using gradient boosting

machine with feature engineering (gbm-wfe), Kurdistan Journal of Applied Research 5 (2020) 28–48.
[32] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.
[33] J. Yang, C. Zhao, H. Yu, H. Chen, Use gbdt to predict the stock market, Procedia Computer Science 174 (2020) 161–171.
[34] G. Ridgeway, D. Madigan, T. S. Richardson, Boosting methodology for regression problems, in: Seventh International Workshop on Artificial Intelligence and Statistics, PMLR, 1999.
[35] I. K. Nti, A. F. Adekoya, B. A. Weyori, A comprehensive evaluation of ensemble learning for stock-market prediction, Journal of Big Data 7 (2020) 1–40.
[36] P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Machine learning 63 (2006) 3–42.
[37] S. R. Polamuri, K. Srinivas, A. K. Mohan, Stock market prices prediction using random forest and extra tree regression, International Journal of Recent Technology and Engineering. 8 (2019) 1224–1228. [38] C. Janiesch, P. Zschech, K. Heinrich, Machine learning and deep learning, Electronic Markets 31 (2021) 685–695.
[39] B. Lim, S. Zohren, Time-series forecasting with deep learning: a survey, Philosophical Transactions of the Royal Society A 379 (2021) 20200209.
[40] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780. [41] A. Moghar, M. Hamiche, Stock market prediction using lstm recurrent neural network, in: ANT/EDI40, 2020.
[42] K. Cho, B. Van Merri¨enboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoderdecoder for statistical machine translation, arXiv preprint arXiv:1406.1078 (2014).
[43] W. Lu, J. Li, Y. Li, A. Sun, J. Wang, A cnn-lstm-based model to forecast stock prices, Complexity 2020 (2020).
[44] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.
[45] H. Li, Y. Shen, Y. Zhu, Stock price prediction using attention-based multiinput lstm, in: Asian conference on machine learning, PMLR, 2018, pp. 454–469.
[46] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473 (2014).
[47] T. Chai, R. R. Draxler, Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature, Geoscientific model development 7 (2014) 1247–1250.
[48] N. Naik, B. R. Mohan, Novel stock crisis prediction technique—a study on indian stock market, IEEE Access 9 (2021) 86230–86242.