Machine Learning And Deep Learning Models: Short-Term Stock Price Prediction

Eric Chen
Faculty of Applied Science
Simon Fraser University
Burnaby, Canada
eca103@sfu.ca

Shao-en Hung
Faculty of Applied Science
Simon Fraser University
Burnaby, Canada
shaoenh@sfu.ca

Jun Pin Foo
Faculty of Applied Science
Simon Fraser University
B.C., Canada
junpinf@sfu.ca

***Abstract*—Stock market is one of the most important tools to grow personal wealth. While there is an abundance of articles showing the success of machine learning and deep learning models in long-term prediction, this paper focuses on short-term stock price prediction to help traders gain an edge and facilitate the process.**

Firstly, we did research to find popular models for prediction given data over a period of time. With that, we tested each models utilizing different sets of technical indicators and also different sized time windows along side closing price. Furthermore, we optimized each individual model's parameters and recorded their performance based on R^2 score and mean absolute percentage error. They were then compared against other model to finally conclude the best performed model. Our results found that MLP has the best performance in price prediction, but not only that, deep learning models overall was observed to be better than machine learning models. Although each model scored well, we recognized an inherent problem in short-term stock price prediction where models lack behind the true price trend due to both the nature of these models and the property of time series data.

***Index Terms*—machine learning, deep learning, support vector, random forest, long-short term memory, multilayer perceptron, short-term time-series forecast**

I. Introduction

Financial stability is critical to survival in modern society. As inflation continues to soar in recent years, the value of money continues to depreciate while cost of living climbs ever higher, causing great difficulties for an individual to sustain their quality of living. To grow one's wealth and persevere through financial turmoil, one solution is to turn to the stock market to trade.

Trading securities as an activity has been around for more than two centuries, with the New York Stock Exchange claiming its origin dates back as early as 1792. (NYSE, n.d.) Investors and speculators alike use stock exchanges to claim stakes in a company for monetary compensation, with hopes that the company will grow profitable, leading to a return on investment through dividends or increase in stock valuation.

Trading brings great rewards and comes with great risk. The NASAA reports 70 percent of all traders lose money, and only 11.5 percent of short-term traders are profitable.(NASAA, 1999) Though the report is two decades old, the situation has not changed since. The recent short squeezes of GME and AMC, as well as social media platforms have encouraged the public to trade risky stocks and instruments, often uninformed, causing tremendous losses or even a complete wipe-out of their portfolio.

Therefore, information is of the utmost importance in trading, and disciplined traders apply technical analysis techniques through analysis of price action, candlestick patterns and indicators in an attempt to predict price movements and determine optimal entry and exit points. With most indicators being derived from historical price, which is a type of time series data-set, we have selected one machine learning models and two deep learning model that are considered to perform well in time series forecasting and widely used for similar applications, which are Support Vector Machine, Long-Short Term Memory and Multilayer Perceptron. The random forest also selected as our baseline model.

Our goal for this project is to determine whether machine/deep learning can be used to predict price movements, and compare the precision of different ML/DL models like Support Vector Machine (SVM), Long-Short Term Memory (LSTM) and Multilayer Perceptron (MLP), to decide the best overall model for short term price prediction.

We will use historical price of selected stock(s) as a daily time series acquired from IEX Cloud, a freemium source of market data, to train the aforementioned models, then feed them with recent price data of corresponding stocks and evaluate whether the predictions are precise and provide actionable information for trading by comparing the Mean Absolute Percentage Error of daily price between models.

II. Main Result

After evaluating the predicted short-term stock price generated by all the models, the final result indicates that the Multilayer Perceptron(MLP) has the highest performance in stock prediction. Moreover, the deep learning models seem to have better performance in the predicted stock price in short term compared to the machine learning models. However, we also found that there is an inherent problem of time series prediction where the model's predictions always lag behind the true value due to the autoregressive nature of stock prices (time series data). If there is no time pattern in the data, the model can't predict effectively. Therefore, the stock prediction in short term might not be viable due to the characteristic of time series data.

III. Contributions

This project aims to find the viability of stock prediction in short term and the prediction performance within different machine/deep learning models. The following parts are the conclusion or method we contribute in the short-term stock prediction topic.

- We implemented different common technical indicators and training models with indicators as feature data. The final result indicates that applying indicators does not necessarily lead to improvements. In fact, applying quantitative indicators make the predicted result worse in some machine learning models.
- We also tested the models' prediction performance with the different training window sizes. We observe that the window size is one of the essential factor in prediction. The final result indicates that the different models may have different optimal window sizes w.r.t prediction performance.
- We observe that all the model's prediction lagged behind the validation data. While we attempted tuning hyper-parameters to mitigate this problem, we did not find any success in resolving this issue, suggesting that the models may not be suitable for use in short-term predictions.

IV. Related Works

From our research, we can see that there have been multiple experiments done to test the accuracy of different algorithms, popular ones being SVM, LSTM, MLP etc. Most of them utilize the same format to compare accuracy by using popular indicators such as Bollinger bands and Relative Strength Index (RSI) alongside the price which proves to be more effective at predictions as in one paper, their experiment concluded that adding these financial technical indicators with machine learning algorithms has resulted in better results than other papers that they found (Zheng & Jin, 2017). Furthermore, all of them also make use of a fixed past data-set to train their algorithm which is used to ensure the consistency across all algorithms.

Karmiani, Kazi, A. Nambisan, and Kamble (2019) had stated that long term predictions are generally accurate. In particular it has an accuracy rate of around 70 percent (Zheng & Jin, 2017). Hence we will avoid that part of prediction and focus on more short term prediction such as the day after. The most common way of short term prediction takes into consideration time series method which considers data over a period of time and trend extrapolation method which uses popular indicators as stated above to make judgment (Z & J, 2020). This mainly converts the stock price into a time series, and predicts the stock price through a time model, and then figures out the fluctuation state of the stock market price.

In general, all methods of analysis determine the stock prices based on the past performance value of the stock and with the help of indicators such as RSI and Bollinger bands. In one research, the performances of a particular stock is compared to understand whether the additional external input indicators provide enhancement to the performance of the base SVM and LSTM models on the original data, and it was found that it does improve SVM and LSTM because of the smoothing effect of the moving averages on the data which helps in learning the influence of the external indicators (Lakshminarayanan & McCrae, 2019).The only problem when using these indicators is that they are mostly applicable for a specific time frame only (Gurav & Sidnal, 2019).Because, stock price is determined by multiple factors which are volatile in nature, therefore it is not easily predictable hence the low accuracy rate.

Lastly, other than using indicators to aid in the process of prediction, some papers have taken sentiment analysis into consideration as well. It was observed that factors such as peoples' sentiments and current occurring events affect the stock market, for e.g. national elections (Singh, Madan, Kumar, & Singh, 2019). In the paper, SVM is used to predict the sentiment of each data and then classifying the tweets into positive and negative tweets for easier and better prediction models. The results had an accuracy of around 76.6 percent (Singh et al., 2019).

In conclusion, we have decided to incorporate all the methods that have been used in the papers discussed above, which is utilizing base predictive models such as SVM, LSTM and back propagation alongside indicators such as RSI, moving average, and Bollinger bands and sentiment analysis to hopefully provide a precise and accurate result.

V. Methodology

Stock price prediction is a very difficult problem to solve. The nature of the stock market means that there are uncountably many variables determining the price of individual stocks at any given time. The three aforementioned models are some of the most commonly used for this scenario, and have particular characteristics that makes them suitable for the task at hand.

A. Support Vector Machine (SVM)

This is a popular algorithm used for many classification and regression problems (Zheng & Jin, 2017). This is a supervised learning model that is based on the statistical learning frameworks or VC theory proposed by (Cortes & Vapnik, 1995). SVM works by mapping training examples to points in space as to maximize the width of the gap between the two categories as seen in Fig 1 (Cortes & Vapnik, 1995). When new examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. SVM is more effective in higher dimensions, this is because of the presence of hyperparameters such as regularization parameter (c) (Karmiani et al., 2019), c will decide how much misclassification of data is allowed.

The hyperplane is the set of points x satisfying

$$w^T x - b = 0$$

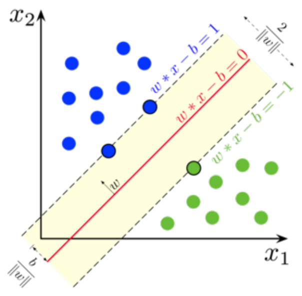

Fig. 1. SVM margin by Cortes, Vapnik.

Where w is the normal vector to the hyperplane. The distance of vector w from origin to decision boundary is 'c'.

$$\vec{x}\vec{w} = 0 \text{ (point lies on the decision boundary)}$$

$$\vec{x}\vec{w} > c \text{ (positive sample)}$$

$$\vec{x}\vec{w} < c \text{ (negative sample)}$$

hence , we can define a decision rule

$$y = +1 \text{ if } \vec{x}\vec{w} + b >= 0$$

$$y = -1 \text{ if } \vec{x}\vec{w} + b < 0$$

RBF kernel:

$$f(x1, x2) = e^{\frac{-||x1-x2||^2}{2\sigma^2}} \quad (1)$$

σ is the variance and our hyperparameter, and $||x1 - x2||$ is the distance between 2 points. The reason for choosing SVM with RBF is because it works great for classification and regression and will minimize the mis-classification error. It works by mapping linear input vectors containing price change and indicator data into the high-dimensional feature space. A solution with SVM will therefore be globally optimal as overfitting is unlikely to occur, preventing fringe events from affecting price change predictions without losing generality. Furthermore, we will use the RBF kernel to facilitate the process as it requires no prior knowledge, making it suitable for this use case where market conditions may change and prior conditions may not necessarily hold forever.

B. *Long Short term Memory(LSTM)*

LSTM is a special type of Recurrent neural network (RNN) proposed by Hochreiter and Schmidhuber (1997). LSTM makes predictions based on time series data and it has some mechanism to choose to delete partial input to keep only essential data for future prediction. LSTM unit consists of four components including input gate, output gate, forget gate, and cell state. The input gate can process the current input and previous hidden state (past result) to generate the vector to update the cell state (help to predict number in future). The forget state is responsible for deciding which part of input should be discarded to update the cell state. The cell state will combine the value of the previous cell state, forget gate, and input gate to generate the value to help for future prediction. In the end, the output gate generates the predicted number for the current timestamp by processing the input and hidden state value. (Singhal, 2020)

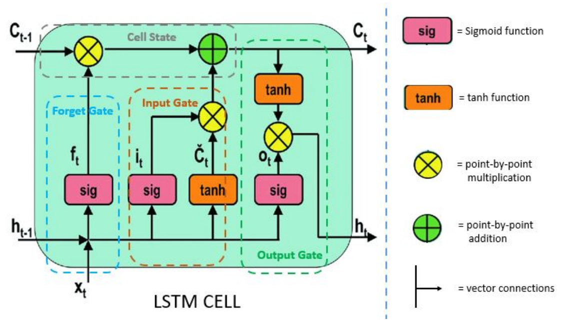

Fig. 2. LSTM unit by Gaurav, Singhal.

$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$
$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$
$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$
$\tilde{c}_t = \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$
$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$
$h_t = o_t \odot \sigma_h(c_t)$
$W, U : weight\ matrix\ for\ different\ gate$
$b : bias\ vector$
$\sigma_g : sigmoid\ function,\ \sigma_h : hyperbolic\ tangent\ function$
$x_t : input\ for\ LSTM,\ f_t : forget\ gate\ result$
$i_t : input\ gate\ result,\ o_t : output\ gate\ result$
$h_t : hidden\ state\ result,\ c_t : Cell\ state\ vector$

LSTM is a more commonly used model when people do stock price prediction. The LSTM contains a different component to make it able to predict the arbitrary number based on past input/output. Moreover, forget gates in LSTM can abandon partial input which can decrease the impact of outliers. It is extremely useful in price prediction since it can handle the case of price abnormally rising/declining rapidly due to some factor in the real world such as war or natural disaster.

C. *Multilayer Perceptron(MLP)*

MLP is a deep learning model based on artificial neural networks. It is composed of at least three layers of fully connected perceptrons, of which there would be one input layer, one output layer, and at least one hidden layer. The model would take the input time series, propagate it forward, compute the error of the prediction and propagate it backwards, then use the error to update the weights of the network using gradient descent. This process is repeated until the weights converge towards optimum, or the maximum number of iterations is reached.

MLP is a popular model that is widely used in both classification and regression problems, due to its ability to solve complex problems with non-linear boundaries, or where variables have complex relationships with each other. Thus, MLP may be a suitable model to be utilised in stock price prediction, which is notably one of the most complicated prediction problems due to the sheer amount of variables contributing to the movement of the stock price. While this does not necessarily mean that increasing number of features guarantee an increase in prediction accuracy, it is still nonetheless a great benefit as it may be able to establish relations between variables that are not immediately apparent.

D. Data-set and Evaluation Metric

In this project, we will conduct our experiment by training all models using historical prices of SPY as acquired from IEX Cloud API (https://iexcloud.io/docs/). The experiment will use the 4 years stock price as a training data-set and 1 year stock prices as the test set for all algorithms to generate the further predicted price. We will implement cross-signal and quantitative indicators and training all the models with indicators as feature data. The quantitative indicator generate a series of numerical data based on historical prices that is used by traders in practice. Our quantitative indicators include the relative strength index (RSI), moving average(MA), volatility and Stochastic Oscillator. Furthermore, traders often view cross-signals of certain indicators such as MA and Moving Average Convergence Divergence (MACD) as bullish/bearish signals and use them to guide their trading strategies, and therefore they are also used as potential features exposing crossover/crossunder events for the relevant indicator to the model. The metric we use to evaluate the performance of the algorithm is Mean Absolute Percentage Error (MAPE). MAPE is primarily used to check whether the predicted relative price change is accurate, where the model performs better the lesser MAPE value is. We will also use the random forest regressor as our baseline algorithm to do stock price prediction. Any algorithm that has lower performance compared to a random forest regressor would be seen as an algorithm not suitable for stock price prediction.

$$MAPE = \frac{1}{n}\sum_{i=1}^{D}\frac{|x_i - y_i|}{x_i} \tag{2}$$

VI. Results

In accordance with the methodology as outlined in the prior section, we have aggregated and computed performance data for each model using different feature sets. For indicators, there are 4 categories presented for comparison labelled ”None”, ”Cross”, ”Quant” and ”All. While ”None” and ”All” are self-explanatory, ”Cross” indicates only cross-signals are added, and ”Quant” indicates only quantitative indicators are added. Henceforth, performance w.r.t. ML and DL models will refer to their prediction accuracy on the testing data-set, specifically the MAPE of their predictions and is better the lower it is. For every trial group we also conducted pairwise t-tests ($H_0 : \Delta = 0$) to test whether the difference in MAPE is significant and to select the superior model if it is. All trial data and code can be viewed in the GitHub repository github.com/limaniner420/PriceActionPredict-ML-DL

For MLP models, it is observed that not using any technical indicators in as additional features gave the best performance (table I). The best window size for this model is 1-day windows (i.e. using only today’s price as predictor) (table II).

For LSTM models, using the cross-signals as additional

Scope	None	Cross	Quant	All
R^2	0.9676	0.9536	0.9606	0.9482
MAPE	0.0111	0.0126	0.0127	0.0143
n	30	30	30	30

TABLE I
MLP RERFORMANCE USING DIFFERENT INDICATORS

Days	1	3	7	14
R^2	0.9676	0.9536	0.9606	0.9482
MAPE	0.0111	0.0126	0.0127	0.0143
n	30	30	30	30

TABLE II
MLP PERFORMANCE USING DIFFERENT WINDOW SIZES

Scope	None	Cross	Quant	All
R^2	0.9416	0.9479	0.6350	0.8224
MAPE	0.0221	0.01425	0.4410	0.0296
n	30	30	30	30

TABLE III
LSTM PERFORMANCE USING DIFFERENT INDICATORS

Days	1	3	7	14
R^2	0.9452	0.9366	0.9520	0.9498
MAPE	0.0145	0.0157	0.0140	0.0143
n	30	30	30	30

TABLE IV
LSTM PERFORMANCE USING DIFFERENT WINDOW SIZES

Scope	None	Cross	Quant	All
R^2	0.9237	0.9375	-3.7311	-4.4037
MAPE	0.0130	0.0134	0.1275	0.1369
n	30	30	30	30

TABLE V
SVM PERFORMANCE USING DIFFERENT INDICATORS

Days	1	3	7	14
R^2	0.9272	0.0905	-3.3678	-5.3544
MAPE	0.0133	0.0431	0.1168	0.1444
n	30	30	30	30

TABLE VI
SVM PERFORMANCE USING DIFFERENT WINDOW SIZES

Scope	None	Cross	Quant	All
R^2	0.8407	0.9130	0.6334	0.6369
MAPE	0.0218	0.0167	0.04158	0.0428
n	30	30	30	30

TABLE VII
RF PERFORMANCE USING DIFFERENT INDICATORS

Days	1	3	7	14
R^2	0.8001	0.8777	0.8248	0.8689
MAPE	0.0244	0.0190	0.0236	0.0212
n	30	30	30	30

TABLE VIII
RF PERFORMANCE USING DIFFERENT WINDOW SIZES

features gave the best performance (table III). The best window size for this model is 7-day windows (table IV).

Both machine learning models (SVM and RF) produced their best performances using only cross-signals gave the best performance(table V, VII). However, SVM performed the best with a 1-day window (table VI) while RF performed the best with a 3-day window(table VIII).

Finally, we selected the best performing feature sets for each model and compared their performance, with our findings as shown below:

Model	MLP	LSTM	SVM	RF
R^2	0.9681	0.9346	0.9379	0.9189
MAPE	0.0110	0.0163	0.0127	0.0169
n	30	30	30	30

TABLE IX
AVG. PERF. WITH BEST FEATURE SETS

From the above table, it can be seen that MLP models performed the best w.r.t. both metrics. (MAPE = 0.0110, R^2 = 0.9681) while RF models performed the worst. (MAPE = 0.0169, R^2 = 0.9119). Deep learning models (MAPE = 0.0137, R^2 = 0.9514), on average, performed better than machine learning models (MAPE = 0.0148, R^2 = 0.9284). It can also been seen that the baseline model (RF) has been outperformed by all other models in both metrics.

Overall, the ranking of the models' performance became clear after performing t-tests on the trial metrics ($h_0 : \Delta = 0, \alpha = 0.05$). The details is available on our GitHub repository in an Excel file, through the following link: https://github.com/limaniner420/PriceActionPredict-ML-DL/blob/main/aggregate_updated.xlsx

MLP is the best model, having clearly outperformed all other models in both metrics, followed by SVM in the second place, with similar R^2 while having lower MAPE when compared to LSTM, the third place, which is then finally followed by RF in the last.

VII. DISCUSSION

Originally, it was expected that the prediction accuracy will gradually increase as more information is made available to the models, with the assumption that the models will be able to use the additional information and be able to discern the correct price movement for the next time period, both in direction and magnitude. This is important as the stock market is volatile, and the same price movement pattern in different contexts can lead to different outcomes (e.g., a bullish price pattern in a bullish market indicates trend continuation, while the same in a bear market may simply be a dead cat bounce, i.e., the rally will soon reverse into falling prices).

However, our testing revealed results to the contrary, with no models having their optimal feature sets being the set with most features (all indicators applied with 14-day window) according t. In fact, MLP performed the best overall, with its optimal feature set being having minimum features (no indicators with 1-day window).

A. Technical Indicators

It does not appear to be the case where increasing the amount of indicators lead to increase in performance. With the exception of MLP as mentioned before, all other models performed the best when only cross-signals are applied. A potential cause can be that the model the quantitative indicators increased the data noise, while in reality traders are mostly concerned only where significant events happen (e.g. short-term MA crossing over long-term MA), which is the primary idea behind cross-signal feature set to emulate ways how real trader uses these indicators.

B. Window Size

We observed mixed results in the effects of window size across different models. Some models performed better with 1-day windows (MLP, SVM) while others performed better with longer window sizes (LSTM: 7-day, RF: 3-day). LSTM performed the best with one of the longer window sizes, possibly because of its ability to "forget" where it is able to partially mitigate the noise from inputs, giving the model greater mileage on longer window sizes than other models that lack this capability. However it should be noted that MLP ultimately outperformed LSTM metric-wise, despite lacking the ability to do so.

C. Model Prediction Behaviour

We have also observed a phenomenon where the the best predictions often mimic and trailed the pattern of the validation data-set by some time-period. While we were not able to ascertain the exact reason for the cause, it is problematic regardless as it effectively meant that the best prediction for tomorrow's price is to apply the difference from yesterday to today's prediction. Given the volatile nature of the stock market, it is very common for stock prices to experience whipsaw (i.e. sudden change in direction) especially in shorter time frames. While long-term investing allows for the price trend to stablise and is relatively insulated against the whipsaw effect, short-term trading using these predictions may lead to immense losses as timing is of paramount importance.

In the case of overall best performing model (MLP, no indicators, 1-day) (fig. 3), the predictions can be seen to completely mimic the validation data with lag after the first day. This suggests that while the models have given impressive metrics on paper, the actual information it presents is inactionable in practice. It is likely that there has not been sufficient preprocessing and preparation of the data to mitigate the

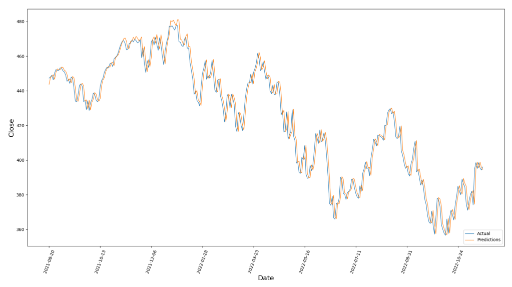

Fig. 3. An example prediction on SPY using MLP with no indicators and 1-day window, versus the validation data

curse of dimensionality that led to this predicament, causing overfitting of the features and their relations.

As a result, while we have determined the best model in terms of evaluation metrics, regrettably we cannot conclude that any of the models will be able to provide any meaningful information or improvement in actual trading scenarios.

D. Limitations

There have been many limitations encountered throughout the project, which prevented us from fully utilising the capabilities of the models.

It should be noted that our relatively basic financial knowledge limited the usefulness as well as the amount of technical indicators that can be applied as additional features. It is clear that while deriving cross-signals has led to improved performance on certain models, it prevented us from confidently identifying the cause of the lagging predictions. We reckon further expanding our knowledge will enable us to generate better, more informative feature sets, which will allow us to better distinguish whether the lagging predictions is inherent to these machine learning and deep learning models, or that it is due to insufficient data preprocessing.

Furthermore, the scope of this project is quite small as we only evaluated performance of 4 models, due to both lack of time and in-depth knowledge regarding machine learning in general. In the future, as we gain more relevant experience and knowledge, we will be better equipped to explore and evaluate more complex or niche machine learning/deep learning models w.r.t their suitability for short-term price prediction.

We seek to improve upon both aforementioned limitations in the future, in order to provide better data preprocessing, evaluation setups, and improve generality and applicability of our findings.

VIII. Concluison

The stock market is one of the way to achieve financial independence, but it also comes with considerable risk. Seeing the lack of short-term prediction research, we decided to choose this area for our study. The goal was to figure out a model that could consistently and effectively predict the price of a stock for a smaller time frame. Whilst considering the garbage-in-garbage-out theory, we focused on different feature sets as input for the training of our models. Each model was trained on two specific categories namely a set of technical indicators and a varying time window of prices, with optimisation of hyperparameters done using Optuna. Finally, their predictions are evaluated using R^2 score and MAPE.

From our comparison, we found that MLP performed the best in price prediction, but not only that, deep learning models were observed to be better than machine learning models. Despite the fact that all model scored well on our test, we discovered an intrinsic problem in short-term stock price prediction where these models shadows the true price trend. We concluded that this was because of the autoregressive nature of stock prices and therefore short-term prediction might not be viable because of the nature of time series data.

In the future, it would be interesting to see if implementing other models could overcome this problem that we encountered. Kim and Han (2000) suggest that artificial neural networks are useful for stock price predictions because a feature discretization model is able to eliminates irrelevant factors, so one of the future directions will be to implement more models based on neural networks. Furthermore, we believe a more comprehensive or larger feature set could perhaps improve the results, at the expense of greater computational complexity. A richer knowledge base might be able to provide a more complex insight on the price trend.

References

Cortes, & Vapnik. (1995). *Support-Vector Networks.* Retrieved from http://image.diku.dk/imagecanon/material/cortes_vapnik95.pdf

Gurav, & Sidnal. (2019). *Predict stock market behavior: role of machine learning algorithms.* Retrieved from https://link.springer.com/chapter/10.1007/978-981-10-7245-1_38/tables/1

Hochreiter, S., & Schmidhuber, J. (1997). *Long Short-Term Memory* (Vol. 9). doi: 10.1162/neco.1997.9.8.1735.

Karmiani, D., Kazi, R., A. Nambisan, A. S., & Kamble, V. (2019). *Comparison of Predictive Algorithms: Backpropagation, SVM, LSTM and Kalman Filter for Stock Market.* doi: 10.1109/AICAI.2019.8701258

Kim, & Han. (2000). *Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index.* Retrieved from http://neuralnetworksanddeeplearning.com/chap2.html

Lakshminarayanan, & McCrae. (2019). *A Comparative Study of SVM and LSTM Deep Learning Algorithms for Stock Market Prediction.* Retrieved from https://www.semanticscholar.org/paper/A-Comparative-Study-of-SVM-and-LSTM-Deep-Learning-Lakshminarayanan-McCrae/fa0cdb970e41ed319a39cf959f40fb3f148e6d6e

NASAA, N. A. S. A. A. (1999). *Report of the day trading project group.* Retrieved from https://

www.nasaa.org/wp-content/uploads/2011/08/NASAA_Day_Trading_Report.pdf

NYSE, N. Y. S. E. (n.d.). *The history of NYSE.* Retrieved from https://www.nyse.com/history-of-nyse

Singh, S., Madan, T. K., Kumar, J., & Singh, A. K. (2019). *Stock Market Forecasting using Machine Learning: Today and Tomorrow.* Retrieved from https://ieeexplore.ieee.org/document/8993160

Singhal, G. (2020). *Introduction to LSTM Units in RNN.* Retrieved from https://www.pluralsight.com/guides/introduction-to-lstm-units-in-rnn

Z, Z. D., Liu, & J, Y. (2020). *Stock Price Prediction Model Based on RBF-SVM Algorithm.* Retrieved from https://ieeexplore.ieee.org/document/9361804

Zheng, & Jin. (2017). *Using AI to Make Predictions on Stock Market.* Retrieved from http://cs229.stanford.edu/proj2017/final-reports/5212256.pdf

Appendix

All values are rounded to 4 decimal places.

Model	MLP	RF	SVM	LSTM
S. Mean	0.0110	0.0169	0.0127	0.0163
S. Variance	7.0898E-08	4.9809E-35	7.7826E-35	3.0738E-06
n	30	30	30	30

TABLE X
BEST MODELS: SUMMARY STATISTICS (MAPE)

Pair	MLP v. RF	MLP v. SVM	MLP v. LSTM	RF v. SVM	RF v. LSTM	SVM v. LSTM
t	-119.9249	-34.2068	-16.1837	2.0202E+15	1.8439	-11.1742
t Crit. (2-tail)	2.0452	2.0452	2.0423	2.0040	2.0452	2.0452
h_0 rej.?	Rej., t <-tCrit	Rej., t <-tCrit	Rej., t <-tCrit	Rej., t >tCrit	No rej., -tCrit <t <tCrit	Rej, t <-tCrit
Best	MLP	MLP	MLP	SVM	n/a	SVM

TABLE XI
BEST MODELS: PAIRWISE T-TESTS (MAPE), $h_0 : \Delta = 0, h_A : \Delta \neq 0, \alpha = 0.05$

Model	MLP	RF	SVM	LSTM
Mean	0.9681	0.9189	0.9379	0.9346
Variance	1.9799E-06	0	1.2751E-32	2.3461E-04
n	30	30	30	30

TABLE XII
BEST MODELS: SUMMARY STATISTICS (R^2)

Pair	MLP v. RF	MLP v. SVM	MLP v. LSTM	RF v. SVM	RF v. LSTM	SVM v. LSTM
t	191.7143	117.7385	11.9348	-9.2181	-5.6270	1.1688
t Crit. (2-tail)	2.0452	2.0452	2.0452	2.0452	2.0452	2.0452
h_0 rej.?	Rej., t >tCrit	Rej., t >tCrit	Rej., t >tCrit	Rej., t <-tCrit	rej., t <-tCrit	No Rej, -tCrit<t <tCrit
Best	MLP	MLP	MLP	SVM	LSTM	n/a

TABLE XIII
BEST MODELS: PAIRWISE T-TESTS (R^2), $h_0 : \Delta = 0, h_A : \Delta \neq 0, \alpha = 0.05$