Stock Market Prediction using Long Short-Term Memory

Shubhodh Amaravadi[1], Pulluri Anudeep[2], Yeramalla Uttam[3], K Nagendra Chary[4]

[1,2,3]B. Tech Scholars, Department of Computer Science and Engineering, SNIST, Hyderabad-501301, India
[4]Assistant Professor, Department of Computer Science and Engineering, SNIST, Hyderabad-501301, India

---

Abstract - *It has never been straightforward to invest in a portfolio of assets since the abnormalities of the financial market prevent simple models from accurately forecasting future asset values. The current main trend in scientific study is machine learning, which consists of teaching computers to execute activities that ordinarily need human intelligence. In order to forecast future stock market values, this article uses recurrent neural networks (RNN), particularly the Long-Short Term Memory model (LSTM). This paper's major goal is to determine how accurately a machine learning algorithm can predict and how much epochs can enhance our model. Time series data are used to represent stock prices, and neural networks are trained to identify patterns from trends in the historical data. To increase the accuracy of stock price prediction, this system used an LSTM algorithm.*

Key Words: Machine Learning, RNN, LSTM, Price Prediction, SVM, Regression.

1.INTRODUCTION

Several research have investigated the use of machine learning in the quantitative financial sector. Machine learning algorithms can be used to predict prices for managing and restricting a full portfolio of assets, as well as for the investing process and many other processes. Machine learning, in its broadest sense, refers to all algorithmic techniques that enable computers to identify patterns based solely on data and without the use of programming instructions. Many models provide a variety of techniques that can be combined with machine learning in quantitative finance, particularly when choosing assets, to predict future asset value. This class of models provides a technique that combines weak sources of information to create an odd tool that can be applied effectively. A critical neural network, gradient boosted regression trees, support vector machines, and random forecast are just a few of the machine learning methods that have recently benefited from the integration of statistics and learning models. These algorithms can make complicated patterns with nonlinear properties and certain relations that are challenging to find with linear methods. Additionally, compared to linear regression methods, these ones demonstrate greater efficacy and multicollinearity. The application of machine learning techniques in finance is the topic of numerous research some of which utilised tree-based models to forecast portfolio returns and others which employed deep learning to create future. The model created by the authors uses the support vector machine (SVM) method and the mean variance (MV) method for portfolio selection. Others then go on to forecast stock returns using this unique decision-making model for day trading investments on the stock market. Deep learning techniques for smart indexing were discussed in another work. In addition, numerous studies have examined a wide range of trends and Applications of Machine Learning in Quantitative Finance. The literature review covered in this paper includes return forecasting, portfolio construction, ethics, fraud detection, decision making, language processing, and sentiment analysis. Since these models don't rely on long-term memory (passed-down data sequences), a class of machine learning techniques based on recurrent neural networks has shown to be particularly beneficial in financial market price prediction. The vanishing gradient problem, which arises as a result of the RNN blocks' repetitive usage of the same parameters at each step, is one of the primary problems with RNN.

While generalising variable-length sequences and maintaining a fixed number of learnable parameters overall, we introduce unique parameters at each step. Internal variables known as Gates are stored in gated cells. Every time step's information, including early states, determines the value of each gate. The many variables of interest are then multiplied by the gate's value in order to affect them. Data collected in a time-series format over a period allows us to track changes over time. Time-series data can monitor development over a period of seconds, days, or even years. This overall increases the accuracy in predicting the stock price.

2. LITERATURE SURVEY

The stock exchange has grown to be one of the most important events in today's financial world. The current state of the stock market has a significant impact on the global economy. People from many walks of life, whether they come from business or academic backgrounds, have been drawn to the stock market with great success. The stock market's nonlinear character has made research on it one of the most important and popular topics worldwide. People choose to invest in the stock market based on their predictions or knowledge from earlier studies. In terms of forecasting, people frequently seek out instruments or strategies that would reduce their risks and maximize their earnings; as a result, stock price forecasting assumes a significant position in the always competitive stock market industry.

Adopting conventional methods like fundamental and technical analysis does not seem to guarantee the predictability's consistency and accuracy. As a result, machine learning technologies have emerged as the most recent trend in stock market forecasting, with predictions based on current market values as a result of training on earlier values. In order to forecast the present trend of the stock market, this article focuses on RNN (Recurrent Neural Networks) and LSTM (Long Short-Term Memory) technology.

3. DESIGN AND IMPLEMENTATION

We will use Python machine learning to estimate stock value using the LSTM.

3.1 Importing the Libraries

Python was used in this study to predict Stock Market Prediction. Several packages have used it, notably Keras and Matplotlib. The commands listed below can be used to install the necessary libraries:

Pip install Keras

Pip install matplotlib

Pip install numpy

Pip install pandas

Keras:

Stock Price Prediction Using a Keras Long Short-Term Memory (LSTM) Model because they have the capacity to store historical data, LSTMs are particularly effective at solving sequence prediction issues. This is significant in our situation since a stock's historical price plays a key role in determining its future price.

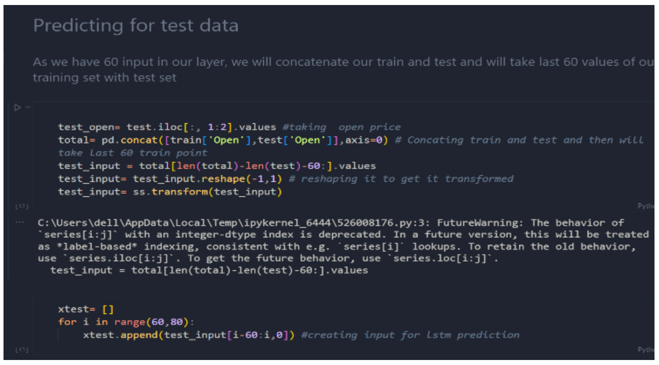
Matplotlib:

A low-level Python package used for data visualisation is called Matplotlib. It emulates MATLAB-like graphs and visualisations and is simple to use. This library's plots, which include line charts, bar charts, histograms, and more, are constructed on top of NumPy arrays.

Numpy:

Python numpy library is used in stock market prediction for performing some scientific calculations all around.

We are predicting the test data using transform function and providing the epoch values around to 50 and then providing the range of -1 to 1 to the reshape function. In the transform function the parameter provided is the test_input.



Predicting for test data

As we have 60 input in our layer, we will concatenate our train and test and will take last 60 values of ou training set with test set

```
test_open= test.iloc[:, 1:2].values #taking  open price
total= pd.concat([train['Open'],test['Open']],axis=0) # Concating train and test and then will take last 60 train point
test_input = total[len(total)-len(test)-60:].values
test_input= test_input.reshape(-1,1) # reshaping it to get it transformed
test_input= ss.transform(test_input)
```

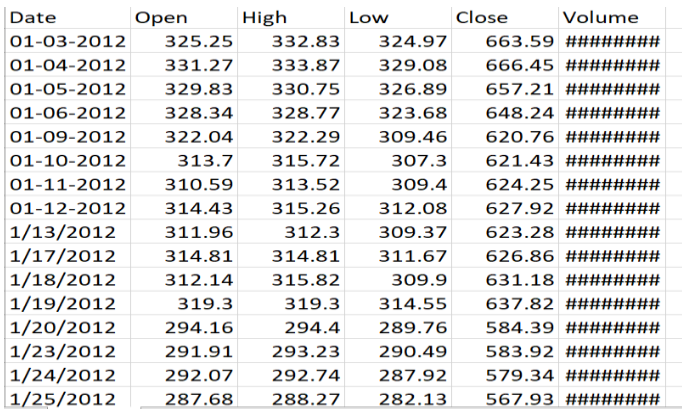
```
C:\Users\dell\AppData\Local\Temp\ipykernel_6444\526008176.py:3: FutureWarning: The behavior of
`series[i:j]` with an integer-dtype index is deprecated. In a future version, this will be treated
as *label-based* indexing, consistent with e.g. `series[i]` lookups. To retain the old behavior,
use `series.iloc[i:j]`. To get the future behavior, use `series.loc[i:j]`.
  test_input = total[len(total)-len(test)-60:].values
```

```
xtest= []
for i in range(60,80):
    xtest.append(test_input[i-60:i,0]) #creating input for lstm prediction
```

Fig -1: Predicting the test data

3.1 Dataset

For the purpose of data visualization, the data is taken from the sample datasets and stored in the csv file.



Date	Open	High	Low	Close	Volume
01-03-2012	325.25	332.83	324.97	663.59	########
01-04-2012	331.27	333.87	329.08	666.45	########
01-05-2012	329.83	330.75	326.89	657.21	########
01-06-2012	328.34	328.77	323.68	648.24	########
01-09-2012	322.04	322.29	309.46	620.76	########
01-10-2012	313.7	315.72	307.3	621.43	########
01-11-2012	310.59	313.52	309.4	624.25	########
01-12-2012	314.43	315.26	312.08	627.92	########
1/13/2012	311.96	312.3	309.37	623.28	########
1/17/2012	314.81	314.81	311.67	626.86	########
1/18/2012	312.14	315.82	309.9	631.18	########
1/19/2012	319.3	319.3	314.55	637.82	########
1/20/2012	294.16	294.4	289.76	584.39	########
1/23/2012	291.91	293.23	290.49	583.92	########
1/24/2012	292.07	292.74	287.92	579.34	########
1/25/2012	287.68	288.27	282.13	567.93	########

Fig -2: Dataset with Stock Prices

The user can view the data to get more information. This saves the stock prices that are obtained from the test csv file.

4. LONG SHORT-TERM MEMORY (LSTM)

4.1 Python

Python is a preferred programming language because of its extensive capabilities, applicability, and simplicity. Due to its independent platform and widespread use in the programming community, the Python programming language is the most suitable for machine learning.

One of the numerous varieties of recurrent neural network (RNN), long short-term memory (LSTM), is also capable of capturing input from earlier stages and using it to make predictions for the future. An artificial neural network, in general (ANN) comprises the following layers: the input layer, the hidden layer, and the output layer.

The number of nodes in the input layer of a NN with a single hidden layer always depends on the dimension of the data, and the nodes of the input layer are linked to the hidden layer by connections known as "synapses." Every two-node relationship from (input to the hidden layer) contains a coefficient called weight that determines how signals are processed.

The results of this transformation form the output layer of our NN, however, these values might not be the best output, in which case a back propagation process will be used to target the ideal value of error. The back propagation process connects the output layer to the hidden layer, sending a signal that conforms to the best weight with the output values.

The least expensive error across the chosen number of epochs. Repeating this process will help us improve our forecasts and reduce prediction error.

This process will be finished with the model being trained. Recurrent Neural Networks (RNN) are a family of neural networks that predict future values based on previous sequences of observations.

The LSTM is a particular class of RNNs due to its capacity for memorizing data sequences. Each LSTM node must consist of a collection of cells that are responsible for storing previous data streams. The upper line in each cell serves as a transport line connecting the models and transfers data from the past to the present. The sigmoidal neural network layer that makes up the gates ultimately drives the cell to an ideal value by discarding or allowing data to pass.

4.1 Recurrent Neural Network (RNN)

Python The deep learning algorithm family includes recurrent neural networks. Because of the feedback links in its architecture, it is a recurrent network. Because it can analyze the complete sequence of data, it offers an advantage over conventional neural networks. The cell, input gate, output gate, and forget gate are all parts of its architecture.

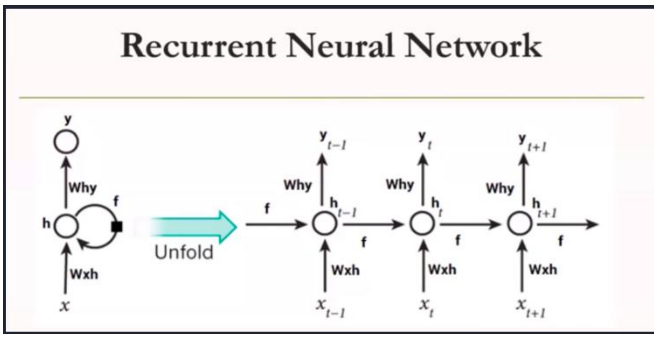


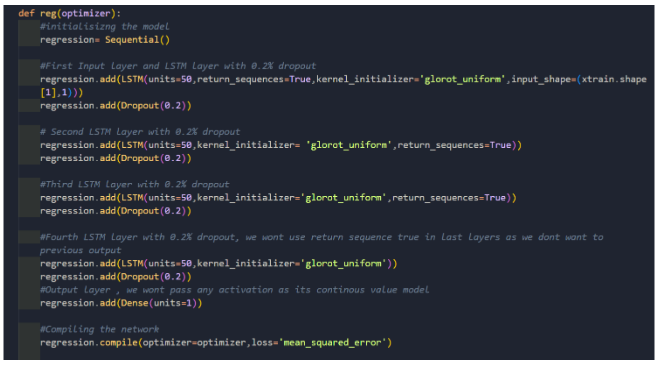
Fig -3: Recurrent Neural Network

We will scale down the stock values to values between 0 and 1 in order to lower the processing cost of the data in the table. By decreasing all the data in big numbers in this manner, memory use is decreased. Additionally, by scaling down, we can get greater accuracy because the data is not widely dispersed. The MinMaxScaler class of the sci-kit-learn library does this.

Text Blob returns a sentence's polarity and subjectivity. The polarity scale is [-1,1], where -1 represents a negative emotion and 1 represents a good emotion. Negative words turn the polarity around. Semantic labels in TextBlob facilitate detailed analysis. Emoticons, exclamation points, emoticons, etc. are a few examples. The range of subjectivity is [0, 1]. Subjectivity measures how much information and subjective opinion are present in the text. The content contains opinion rather than information due to the text's heightened subjectivity. One other setting for TextBlob is intensity. TextBlob uses the "intensity" to determine subjectivity. Whether a word modifies the next word depends on its intensity. Adverbs are used as modifiers in English, such as "extremely good."

4.1 Regression

The Regression-based model relied on a linear function to forecast continuous values based on previously provided values [2]. The general procedure begins with the extraction of input data from stock movement, then the machine learning algorithm is performed to develop the regression-based model that delivers the final forecast of the stock prices, according to Siew and Nordin. The formula is often V = a + bK + error, where V is the projected continuous value, K is the previously supplied value, and a and b are coefficients. Regression takes into account all five variables, and the R-square test was used to determine the confidence level. The goal and benefit of this technique is to reduce the error in the formula.



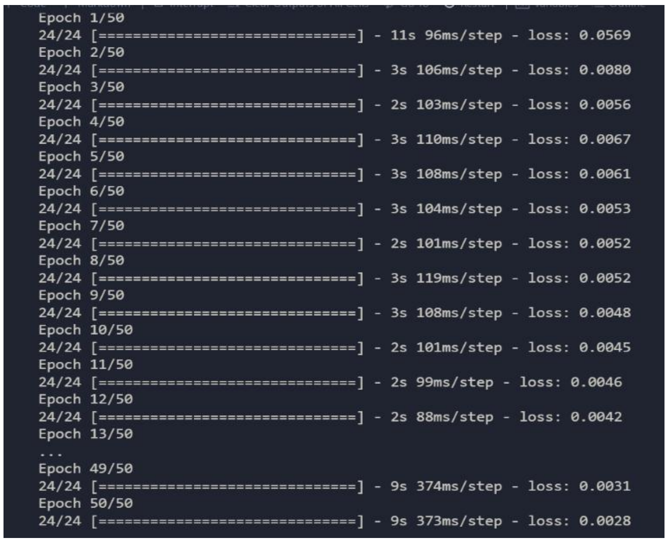
```
def reg(optimizer):
    #initialisizng the model
    regression= Sequential()

    #First Input layer and LSTM layer with 0.2% dropout
    regression.add(LSTM(units=50,return_sequences=True,kernel_initializer='glorot_uniform',input_shape=(xtrain.shape
    [1],1)))
    regression.add(Dropout(0.2))

    # Second LSTM layer with 0.2% dropout
    regression.add(LSTM(units=50,kernel_initializer= 'glorot_uniform',return_sequences=True))
    regression.add(Dropout(0.2))

    #Third LSTM layer with 0.2% dropout
    regression.add(LSTM(units=50,kernel_initializer='glorot_uniform',return_sequences=True))
    regression.add(Dropout(0.2))

    #Fourth LSTM layer with 0.2% dropout, we wont use return sequence true in last layers as we dont want to
    previous output
    regression.add(LSTM(units=50,kernel_initializer='glorot_uniform'))
    regression.add(Dropout(0.2))
    #Output layer , we wont pass any activation as its continous value model
    regression.add(Dense(units=1))

    #Compiling the network
    regression.compile(optimizer=optimizer,loss='mean_squared_error')
```

Fig -4: Regression

The resultant graph of batch size 512 and 90 epochs was formed after performing the linear regression algorithm, and the confidence score was 0.86625. The chance of a correct forecast is 86.625% according to the confidence score. This outcome demonstrates the technique's potential and how it enhanced prediction accuracy

5. RESULT

The portfolios created using the model results are described in this section. Each trading day, 400 equities were available from which to choose and assemble a portfolio. The portfolios in this study were created by ranking each day's predictions for all stocks and then choosing the equities with the highest predictions. As a result, since the targets of LSTM networks and CART models are return rates along with their respective probabilities, the stocks are simply ranked by the professional values.



```
Epoch 1/50
24/24 [==============================] - 11s 96ms/step - loss: 0.0569
Epoch 2/50
24/24 [==============================] - 3s 106ms/step - loss: 0.0080
Epoch 3/50
24/24 [==============================] - 2s 103ms/step - loss: 0.0056
Epoch 4/50
24/24 [==============================] - 3s 110ms/step - loss: 0.0067
Epoch 5/50
24/24 [==============================] - 3s 108ms/step - loss: 0.0061
Epoch 6/50
24/24 [==============================] - 3s 104ms/step - loss: 0.0053
Epoch 7/50
24/24 [==============================] - 2s 101ms/step - loss: 0.0052
Epoch 8/50
24/24 [==============================] - 3s 119ms/step - loss: 0.0052
Epoch 9/50
24/24 [==============================] - 3s 108ms/step - loss: 0.0048
Epoch 10/50
24/24 [==============================] - 2s 101ms/step - loss: 0.0045
Epoch 11/50
24/24 [==============================] - 2s 99ms/step - loss: 0.0046
Epoch 12/50
24/24 [==============================] - 2s 88ms/step - loss: 0.0042
Epoch 13/50
...
Epoch 49/50
24/24 [==============================] - 9s 374ms/step - loss: 0.0031
Epoch 50/50
24/24 [==============================] - 9s 373ms/step - loss: 0.0028
```

Fig -5: Predicted Stocks After Applying LSTM

A portfolio of equities would not be diversified if too few stocks were chosen. The benefit of selecting the stocks with the highest prediction would disappear if too many stocks were chosen. The overall investment and transaction fees would be excessive if one chose too many equities to trade each day. As a result, an optimal k, or the number of stocks to choose for each stock portfolio, must be established.

The k with the highest portfolio returns on investment was chosen as the optimal k.

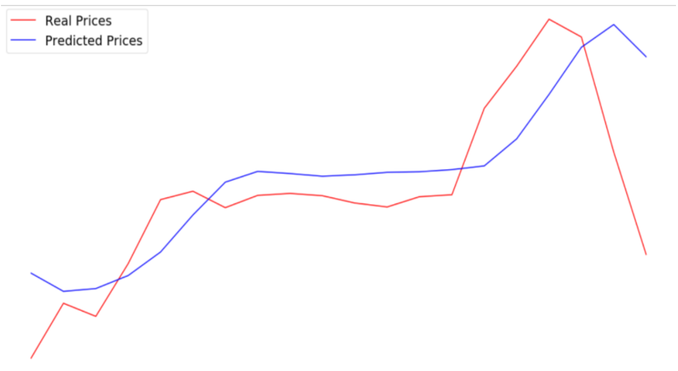


Fig -3: Output of the Stock Market Prediction

6. CONCLUSIONS

The CART methodology is modelled after the structure of a tree and explains the prediction of stock prices, whereas the ARIMA model is one of many conventional methods that rely on an underlying stochastic model to extract information from the data. CART model underperformed the ARIMA model.

The LSTM model assumes that the underlying process is unknown and is an algorithmic method for analyzing time series data. Compared to the CART model in this investigation, our LSTM model was able to forecast stock price fluctuations with a higher degree of accuracy. This is due to the LSTM model's inherent ability to analyze sequential data, extract useful information, and discard irrelevant information using a deep learning technique.

As a result, the portfolio chosen by our LSTM network had a somewhat more consistent return than the ARIMA model. This is because, in contrast to conventional linear statistical models, a deep learning model is better able to extract the nonlinear relationship in the data.

Practically, we recommend using the LSTM model for stock price forecasts due to the deep learning approach of the LSTM model for investors interested in selecting between statistical approaches, machine learning techniques, or deep learning techniques to determine which stocks to buy.

REFERENCES

[1] Huang, W., Nakamori, Y., Wang, S. Y (2005). Forecasting stock market movement direction with support vector machine. Computers & Operations Research. Vol. 32(10), Pages, 2513-2522.

[2] Kim, K-J, Han, I (2000). Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. Expert Systems with Applications, Vol. 19.

[3] Lahmiri, S (2012). Linear and nonlinear dynamic systems in financial time series prediction. Management Science Letters 2, pages, 2551–2556

[4] Levin, N, Zahavi, J (2001). Predictive modelling using segmentation. Journal of Interactive Marketing, Vol. 15.

[5] Wang, S., and Y. Luo. 2012. “Signal Processing: The Rise of the Machines.” Deutsche Bank Quantitative Strategy (5 June).

[6] Takeuchi, L., & Lee, Y. Y. A. (2013). Applying deep learning to enhance momentum trading strategies in stocks. In Technical Report Stanford University.

[7] Siami-Namini, S., & Namin, A. S. (2018). Forecasting economics and financial time series: Arima vs. lstm. arXiv preprint arXiv:1803.06386.

[8] Patterson J., 2017. Deep Learning: A Practitioner’s Approach, O’Reilly Media.

[9] Olah, C. (2015). Understanding lstm networks–Colah’s blog. Colah. GitHub. io.