Stock market prediction using KNN and RNN – LSTM

Chaitanya Gudimalla
MS in Computer Science
University of Central Florida
cg7614@knights.ucf.edu

Ashritha Kuchibhotla
MS in Computer Science
University of Central Florida
ashritha.kuchibhotla@knights.ucf.edu

Sai Sriram Kurapati
MS in Computer Science
University of Central Florida
sriram_k@knights.ucf.edu

***Abstract*— The process of forecasting future stock prices on a stock exchange in order to make money is known as stock market prediction. High precision stock market forecasting has become a difficult undertaking, but machine learning has enabled the creation of many data models that can improve our performance. In this research, the future stock price using data from the "S&P 500" index is predicted using RNN(Recurrent neural networks) and LSTM(Long Short Term Memory) layers along with KNN.. Data generation comes first, then preprocessing, feature selection, training, and outcome comparisons. Different parameters can be used when training the model. The train and test accuracy for KNN are computed for analysis. Mean Absolute is used to predict the LSTM error percentage.**

***Keywords*— S&P 500, LSTM, RNN, *K-nearest neighbor*.**

I. INTRODUCTION

The foundation of any nation's economy is its financial sector. Due to several technology advancements, how business is now performed has an impact on how a nation's economy is produced. According to the World Bank, there are currently 93 trillion US dollars worth of publicly traded enterprises. The stock market is a public venue where shares of corporations with public listings can be bought and sold. Equity, which is another name for stocks, essentially denotes ownership in a business. The stock exchange is where shares are bought and sold. Here are some significant instances where stock markets are crucial:

- Maintaining the health of the economy
- Increasing a company's capital,
- Assisting individuals in investing in quickly expanding businesses for their own benefit.

Because of this, it is highly profitable and desired for all types of investors to develop a system that can educate from the past data and predict the eventual value of a company.

Creating a machine learning model to forecast stock price is the primary goal of this research. To

create the model, by using RNN(Recurrent neural networks) and LSTM(Long Short Term Memory) layers along with KNN. We then compared the output of both models. RNN makes use of earlier phases to learn from historical data and forecast future stock value. The LSTM has this data. This strategy aids in dynamic operation and prediction.

For the creation of both of our models, we used Python. Our project will be helpful for making precise stock price predictions. At the conclusion of this research, we will also contrast the accuracy of KNN and LSTM-RNN.

This essay is divided into six parts, the first of which gives a brief overview of the paper's subject. Section Two is a summary of the literature review, which gives a brief description of the systems used to forecast stock values on a stock market. Our research focuses on time series analysis, graph-based techniques, deep learning, and neural networks.

The workings of machine learning algorithms such as RNNs and LSTMs, KNN, and Time Series Analysis are described and provided with an overview in Section III THEORETICAL AND CONCEPTUAL STUDY. There is also a quick explanation of the Mean Absolute Error and the Root Mean Square Error. Section IV, "Dataset and Preparing," describes our methods for preprocessing datasets. Section V, "RESULTS AND ANALYSIS," has a table with a definition of our model and the results of our algorithm. Section CONCLUSION AND Further WORK, which includes makes recommendations for future research, concludes the paper.

II. BACKGROUND WORK

Data is categorised based on similarity of feature, which is determined by well-known methods like hamming distance, euclidean distance etc., when different algorithms such as KNN are applied to the data that is gathered. The K-most comparable characteristics are chosen, and the class which is most prevalent is given as the class for unused and fresh test data.

When TSA is used in conjunction with RNN and LSTM, the model is significantly more capable of forecasting future values from the available current and past data. The most recent information and previous information can naturally be connected by RNN before selecting what the result should be. However, the drawback of relying solely on RNN is that it can only connect data as time goes on and can only preserve knowledge from the recent past. As a result, LSTM and RNN are coupled to produce superior forecasts. In order to create precise predictions, LSTM has memory that can retain information for longer periods of time. This LSTM approach makes it more effective in forecasting prices of the stock.

III. THEORETICAL ANALYSIS

A. TSA (Time Series Analysis) :

Time series is an assortment of various data points taken across time for a given amount of time. Data points given for the target or response variable, TSA is used to look at the characteristics of the response attribute in relation with time. We can track the factors that affect the variable over time thanks to TSA. TSA assists in constructing a range of time-based analyses and conclusions for problem definition whose goal is

- Intention analysis
- Segmentation
- Forecasting
- descriptive analysis
- classification

As our issue statement is time-based, we will primarily concentrate on employing TSA as the foundation for anticipating future evaluation and prediction in this study.

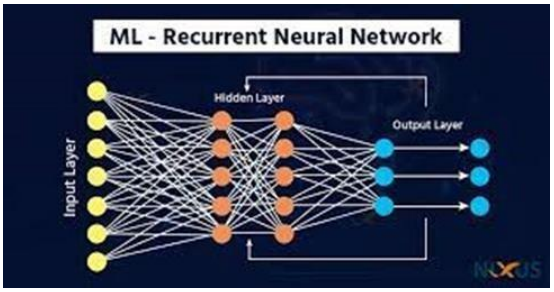
To forecast the price of a stock, TSA will use a single-dimensional input made up of past data. The model will be trained using the dataset, and it will then project future values.

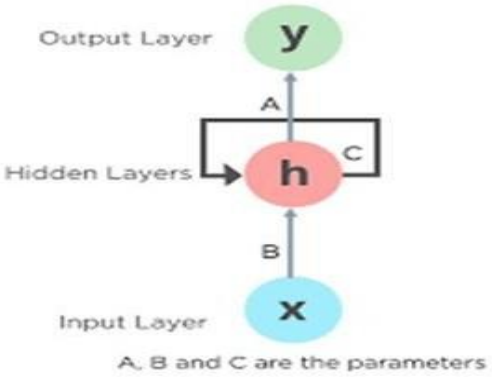
The article "A survey on forecasting of time series data" [1] provides a thorough overview of the many techniques used to anticipate various types of time series dataset. Higher optimization processes and algorithms are employed for better performance and accuracy, in addition to explaining the general forecasting models.

A. Recurrent Neural Networks (RNN) :

For encoding sequence data, recurrent neural networks (RNNs), a family of neural networks, are helpful [8]. Due to their internal memory, RNNs can accurately predict what will happen next by retaining important information from the input they received. As a result, they are the algorithm of choice for time series and other sequential data. We selected RNN as our issue statement because the flow of facts in time series data provides critical data concerning what will follow next.. Unlike feed-forward neural networks, which only allow information to flow in one way and never make touch with the same node twice, RNNs allow information to pass through a loop. The output is chosen while considering the current input and what has been learned from the inputs.

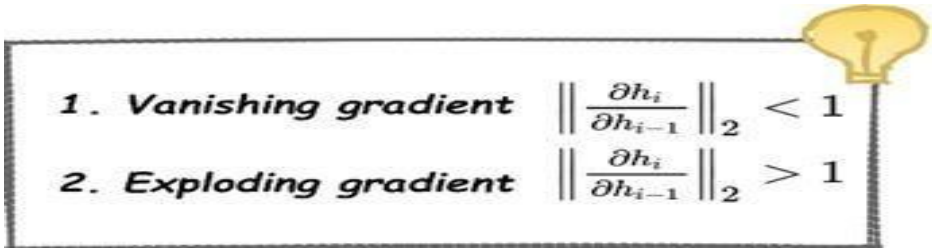
.





.

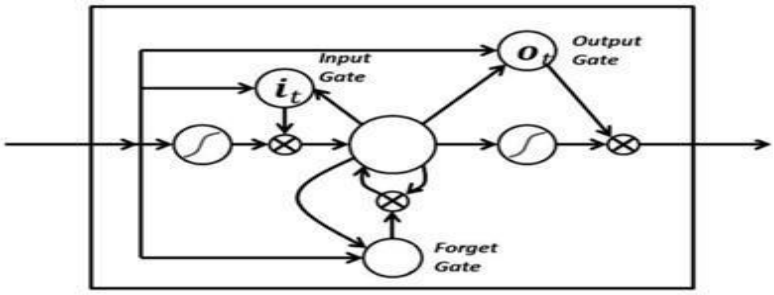
The internal memory of an RNN is what it uses to store the specifics of the data. The output of the RNN is made a copy and re-injected in the system. Weights are used by RNN's to provide the output by applying these weights to both the past and the present source. The vanishing gradients problem and the inflating gradient issue can affect back - propagation algorithm [9] time-series RNNs.. While the gradient is diminishing, it is difficult to understand some long-term dependencies since the term is rapidly approaching zero. Exploding gradients have terms that exponentially approach infinity, leading to an unstable process that causes their value to be NaN. One approach to tackling the problem is to employ the Long Short Term memory algorithm.



B. Long – Short Term memory architecture (LSTM) :

Long Short-Term Memory (LSTM) networks are RNNs with larger memory capacities. By allocating data "weights," LSTMs allow Recurrent neural networks to be open to new knowledge, ignore it, or give it enough weight to change the outcome. The gated cell will decide whether to keep the information or discard it. The importance of the information is determined by the weights that the algorithm assigns. As a result, the algorithm gradually establishes the significance of the information.

Architecture for LSTM: It is built by combining input, forget, and output gates.

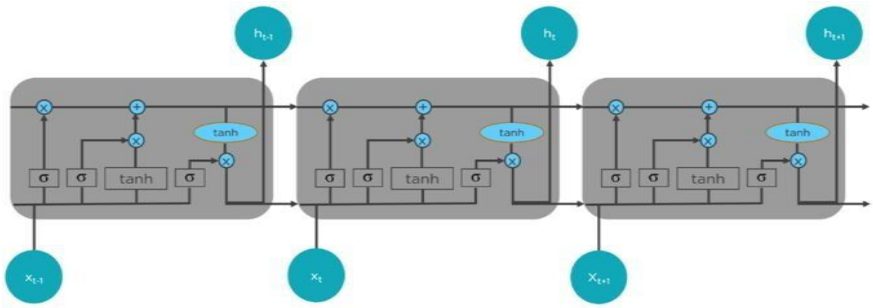


Input Gate: A cell's input gate decides whether or not to include a new input and whether or not it should try to acquire new knowledge from its information. This gate serves two functions. The first is the sigmoid function, and the second is the tanh function. The sigmoid function determines which values are permitted through (1 or 0). The function tanh explains the significance and ranking of these values that are passed from the sigmoid function.. The significance range lies in between (-1,1).

Forget Gate: This feature deletes data if it is unimportant or irrelevant and comes from a previous timestamp. A sigmoid function aids in the decision-making process.

Output gate: The cell must send the modified information from the present timeframe to the future timeframe., which is known as letting the data alter the state's output. The final output value is determined by multiplying the results of the sigmoid and tan functions.

The LSTM's gates' analog sigmoid shape allows for backpropagation. Training requires less time and has a high degree of precision because LSTM maintains slopes that are adequately steep.



B. K-Nearest neighbors (KNN) :

Using K nearest neighbors, a supervised learning method, it is possible to predict the category or continuous value of a new data point. KNN classifies the latest data point to that of the closest training set data point(s) using the "feature similarity" method. Since it doesn't make any prior assumptions about the data, K-nearest neighbor is often referred to as a lazy learning algorithm and non-parametric algorithm because it uses all of the data for training.

The test and train data are loaded first, and then a random integer is selected as the K value. We take the following actions for each test data point:

- The gap between each row in the training data and the test is measured using well-known methods like Euclidean or hamming.

- The next step is to select the K closest instances to the test data point.
- The test data is then assigned to the most prevalent class from the aforementioned K instances.

The initial step in solving the stock prediction problem with KNN is to use the whole training set as the model for the lazy algorithm that is KNN. The prediction is carried out for a brand-new, unknown instance using a full dataset scan and retrieval of the k-most relevant cases.
Here, the Euclidean distance is used to determine how comparable the features are.

C. Root Mean Square Error (RMSE) :

The root mean square error is used to assess the precision of forecasts. It demonstrates the Euclidean separation between forecasts and observed true values. This method is frequently used to assess how accurately a model predicts quantitative data.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2}{N}},$$

D. Mean Absolute Error (MAE):

The mean absolute error for a collection of observations is calculated by averaging the absolute difference between each observation's prediction and actual value. With the use of MAE, users can convert learning challenges into optimization issues.

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n} |y_j - \hat{y}_j|$$

IV. DATASET AND PREPROCESSING

The selected dataset is offered in a csv file format for both KNN and LSTM. For the S&P 500, it was collected via Kaggle Stock Market.
The dataset includes several stock rates, and data is organized under the columns "open," "date," "close," "high," and "low". "high-low" and "Close-open" are two brand-new columns that have been generated and added to the dataset. Close column value for that particular row is subtracted from the value of the open column to form the close-open column. Similar to this, the high-low column is produced by deducting the high column value for that particular row from the

the low column value. The data are gathered daily. The dataset only contained a small numberof NA values and these were dropped to avoid errors.

V. RESULTS AND ANALYSIS

A. Activation functions:

The sigmoid function is used to calculate the outputs for the input, output, and forget gates of an LSTM. The tank activation function is applied to compute cell candidates and hidden states.

B. Error/Loss functions:

Mean Absolute Error (MAE) and Squared Error (RMSE) are the metrics which are being used to analyse the performance of the trained model. How far forecasts deviate from observed real values will be shown by the RMSE.. The accuracy of our model is also determinedusing MAE.

C. Window size:

The model employs windows of different sizes (10, 15, and 20), and it has been found that networks perform better when the size is 15 for LSTM.

D. Number of hidden neurons:

When adjusting the parameters, it was found that the model operates best when there are between 15 and 20 hidden neurons. Additionally, it was found that adding more hidden layers had no discernible impact on how the error functions worked.

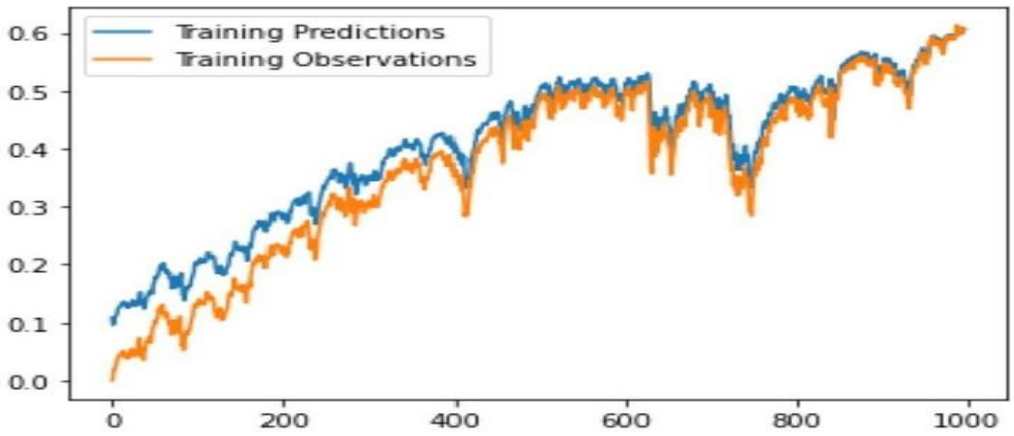
E. Number of epochs:

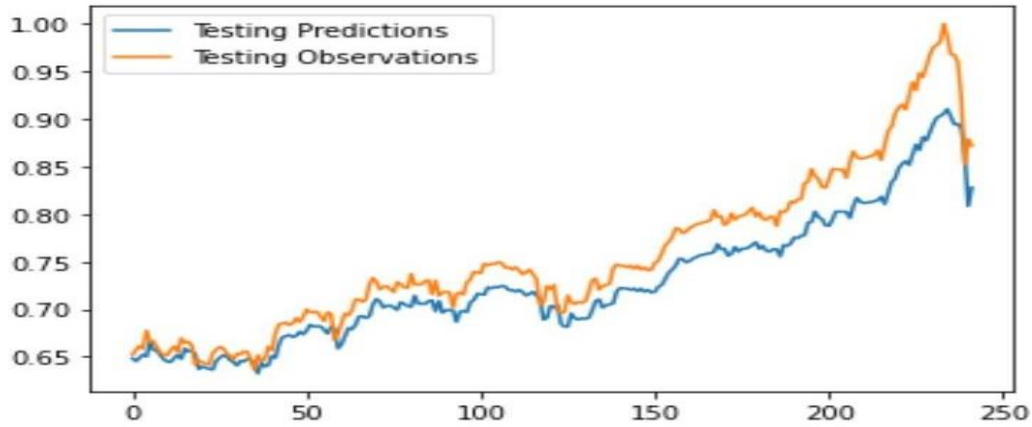
When adjusting the settings, it was discovered that using 400 epochs results in improved predictions of values. The cost-cutoff trade is not significantly altered by increasing epoch's value.

The stock market is so volatile that it is difficult to forecast the exact pattern that the stock prices will most likely follow.

The outcomes of various tests on our model are shown in the table below. Additionally, charts of actual stock market values and predicted prices using the model.

The accuracy on training and testing data for LSTM is shown in the following chart.





Experimental Results:

Index No.	Attributes with values	Accuracies with KNN	Results with LSTM
1	Input value = 10 Output value = 1 Value of epochs = 400 Value of hidden neuron = 15 Rate of Learning = 0.1 K-value=7	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0437 RMSE value for the Testing data = 0.0426 MAE value for the Training data = 0.0374 MAE value for the Testing data = 0.0355 Value of Time taken = 10.89 mins

2	Input value = 10 Output value = 1 Value of epochs = 200 Value of hidden neuron = 15 Rate of Learning = 0.1 K-value=10	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0552 RMSE value for the Testing data = 0.0256 MAE value for the Training data = 0.0435 MAE value for the Testing data = 0.02153 Value of Time taken = 10.87mins
3	Input value = 10 Output value = 1 Value of epochs = 200 Value of hidden neurons = 15 Rate of Learning = 0.05 K-value=50	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0445 RMSE value for the Testing data =0.0374 MAE value for the Training data = 0.0368 MAE value for the Testing data = 0.0295 Value of Time taken = 10.81mins
4	Input value = 10 Output value = 1 Value of epochs = 200 Value of hidden neurons = 15 Rate of Learning = 0.2 K-value=500	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0436 RMSE value for the Testing data = 0.0338 MAE value for the Training data =0.0361 MAE value for the Testing data = 0.0282 Value of Time taken = 10.85 mins

5	Input value = 10 Output value = 1 Value of epochs = 400 Value of hidden neurons = 15 Rate of Learning = 0.2 K-value=200	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0414 RMSE value for the Testing data = 0.0619 MAE value for the Training data =0.0378 MAE value for the Testing data = 0.0519 Value of Time taken = 10.85 mins
6	Input value = 15 Output value = 1 Value of epochs = 400 Value of hidden neurons = 20 Rate of Learning = 0.01 K-value=75	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0538 RMSE value for the Testing data = 0.0380 MAE value for the Training data =0.0449 MAE value for the Testing data = 0.0313 Value of Time taken = 10.82 mins
7	Input value = 10 Output value = 1 Value of epochs = 200 Value of hidden neurons = 20 Rate of Learning = 0.2 K-value=2000	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0594 RMSE value for the Testing data = 0.2459 MAE value for the Training data =0.0480 MAE value for the Testing data = 0.0952 Value of Time taken = 11mins

8	Input value = 20 Output value = 1 Value of epochs = 600 Value of hidden neurons = 20 Rate of Learning = 0.05 K-value=100	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0591 RMSE value for the Testing data =0.0475 MAE value for the Training data = 0.0501 MAE value for the Testing data = 0.0405 Value of Time taken = 10.86 mins
9	Input value = 20 Output value = 1 Value of epochs = 600 Value of hidden neurons = 20 Rate of Learning = 0.2 K-value=50	Accuracy of test data = 0.55 Accuracy of train data = 0.64	RMSE value for the Training data = 0.0527 RMSE value for the Testing data =0.0505 MAE value for the Training data = 0.0425 MAE value for the Testing data = 0.0313 Value of Time taken = 10.87 mins

V.1. Result table

The outcomes of our experiments with both of our models, KNN and LSTM-RNN, using various parameter combinations are displayed in the table below.

VI. CONCLUSION

According to the results of our experiment, KNN model accuracy is approximately 64%, whereas LSTM-RNN model accuracy is greater than 90%. As a result, due to its high degree of dynamicity, LSTM-RNN is better suited for stock market prediction. As the stock market is never steady, it has also been noted that there are now significant changes in the data. Along with their spatial data, if other affecting data is added to the dataset better and accurate predictions can be generated in the future using the current data.

REFERENCES

1) Mahalakshmi, G. et al. “A survey on forecasting of time series data.” 2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16) (2016): 1-8.

2) R. S. Latha et al., "Stock Movement Prediction using KNN Machine Learning Algorithm," 2022 International Conference on Computer Communication and Informatics (ICCCI), 2022, pp. 1-5, doi: 10.1109/ICCCI54379.2022.9740781.

3) San-hong Liu and Fang Zhou, "On stock prediction based on KNN-ANN algorithm," 2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), 2010, pp. 310-312, doi: 10.1109/BICTA.2010.5645310.

4) Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras (Long Short-Term Memory Networks Archives - MachineLearningMastery.com)

5) Multivariate Time Series Forecasting with LSTMs in Keras (https://machinelearningmastery.com/category/deep-learning-time-series/)

6) LSTMs Explained: A Complete, Technically Accurate, Conceptual Guide with Keras(In Medium.com)

7) Z. K. Lawal, H. Yassin and R. Y. Zakari, "Stock Market Prediction using Supervised Machine Learning Techniques: An Overview," 2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), 2020, pp. 1-6, doi: 10.1109/CSDE50874.2020.9411609.

8) http://colah.github.io/posts/2015-08-Understanding-LSTMs/

9) Predicting weather using LSTM (https://www.rs- online.com/designspark/predicting-weather-using-lstm)

10) BackPropagation in RNN Explained(In Medium.com)

11) https://www.tutorialspoint.com/machine_learning_tutorials.htm

12) https://builtin.com/data-science

The link for the code is :

https://drive.google.com/file/d/1LWf27HATHzWVmsDsEKH3rQ3KaFe-b66i/view?usp=share_link