Enhancing Stock Market Predictions with Hybrid Deep Learning

A Report for **B.Tech Project-I**

Submitted in the partial fulfilment of the Degree of Bachelor of Technology

(Mathematics and Computing)

Submitted By:

Priyanshu Barnwal (2K21/MC/127)

Vishwa Doshi (2K21/MC/179)

Jatin Gupta (2K21/MC/78)

Under the Supervision of

Dr. Dinesh Udar

DEPARTMENT OF APPLIED MATHEMATICS

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly, Delhi College of Engineering) Bawana Road, Delhi - 110042

DEPT. OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly, Delhi College of Engineering)

CERTIFICATE

We hereby certify that the project report, ***Enhancing Stock Market Predictions with Hybrid Deep Learning*****,** submitted to Delhi Technological University, Delhi as a partial fulfilment of the requirements for the award of a Bachelor of Technology Degree, is an authentic work completed by us under Dr. Dinesh Udar's supervision.

This submission reflects our thoughts in our own words, and where we have borrowed from others, we have properly and appropriately cited and referenced the original works.

Place: Delhi
Date: 14th December 2024

Approved By:
Dr Dinesh Udar
Dept. of Applied Mathematics
DTU

Submitted By:
Priyanshu Barnwal (2K21/MC/127)
Vishwa Doshi (2K21/MC/179)
Jatin Gupta (2K21/MC/78)

DEPT. OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly, Delhi College of Engineering)

DECLARATION

We hereby certify that the project report titled ***Enhancing Stock Market Predictions with Hybrid Deep Learning,*** submitted to Delhi Technological University, Delhi, as part of the requirements for the Bachelor of Technology Degree, represents our original work carried out under the supervision of **Dr. Dinesh Udar.** This report is a reflection of our own ideas, and all sources of information or content borrowed from others have been duly cited and referenced in accordance with academic standards.

Place: Delhi
Date: 14th December 2024

Submitted By:
Priyanshu Barnwal (2K21/MC/127)
Vishwa Doshi (2K21/MC/179)
Jatin Gupta (2K21/MC/78)

DEPT. OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly, Delhi College of Engineering)

ACKNOWLEDGEMENT

We would like to extend our heartfelt gratitude to everyone who supported us throughout this project, enabling its successful completion. We are deeply thankful to **Prof. R. Srivastava**, Head of the Department of Applied Mathematics at Delhi Technological University, Delhi, for providing the essential resources and guidance needed for this work.

We also wish to express our profound appreciation to **Dr. Dinesh Udar** from the Department of Applied Mathematics, Delhi Technological University, for his invaluable mentorship, unwavering support, and insightful knowledge, which greatly contributed to the success of this project.

Submitted By:
Priyanshu Barnwal (2K21/MC/127)
Vishwa Doshi (2K21/MC/179)
Jatin Gupta (2K21/MC/78)

CONTENTS

INTRODUCTION

Such reflection pillars as economic indicators, corporate performance, geopolitical events, and public sentiment contribute to the uncertainty of financial and economic forecasts in the stock market. This is because it is beyond the traditional approaches of both fundamental and technical analysis into the stock market since those cannot manage the growing complexity and volumes of information.
Deep learning, particularly models of RNNs and LSTMs, holds good potentiality in the stock market. Such models are very much capable of identifying sequential patterns from time-series data, as well as collation sources like historical prices, news, and social media sentiment. This is simply because they fill the gap between the simplicity of tradition and the complexities of the modern market. It leads to a better precision and robust prediction.

Advantages of Deep Learning in Stock Market Analysis:

• Pattern Recognition: It detects complex patterns in large data sets.
• Data Integration: It can process both structured and unstructured data.
• Real-Time Adaptation: It can update itself dynamically with real-time market data.

Promising as it is, deep learning has its own challenges, including computational requirements, overfitting risk, and need for data preprocessing. If addressed, this could make analysts more capable in the handling of financial markets.

MOTIVATION

Stock market analysis is crucial to individual investments and economic policies, and even slight improvements in prediction yield large returns in finance. The increasing complexity and volume of disparate financial data, such as price movements, economic reports, and sentiment, outstrip traditional methods of analysis, including fundamental and technical analysis.

These models, in turn, include Deep Learning models like RNNs and LSTMs, in order to capture sequential dependencies among various data types that reflect the comprehensive view of an analysis. Therefore, they are advanced for processing the financial time series data; it uses past trends, some economic indicators, and sentiment for creating precise predictions.

This research will break through traditional approaches with the use of machine learning in order to create robust, actionable, and timely financial insights that can empower investors and institutions to make better decisions while democratizing financial intelligence within a data-driven economy.

PROBLEM STATEMENT

Predicting the stock market is a very difficult task because of intricate interactions between historical trends, economic indicators, geopolitical events, and market sentiment. Traditional methods, such as technical and fundamental analysis, tend to fail in capturing those complex relationships and nonlinear dynamics behind price movements, especially under conditions of volatility.
This project responds to these challenges by making use of advanced machine learning models for the integration of diverse data sources, such as historical time series and sentiment analysis from news and social media. The objectives are listed below:

- Use of historical time series and other alternative datasets, like sentiment analysis and macroeconomic indicators.
- Advanced predictive models: Deep learning architectures such as RNNs and LSTMs and relational modelling techniques like Graph Neural Networks with transformers capturing relationships among the stocks
- Model explainability so that trust and usability by investors could be achieved
- Back testing of the trading strategies from model output using NIFTY 50 benchmark against which performances can be evaluated.

This project will fill the gap between traditional analysis and modern machine learning by bringing explainability, alternative datasets, and relational modelling together in order to offer enhanced robust and data-driven investment strategies.

LITERATURE REVIEW

1. Traditional stock market analysis techniques

1.1 Technical Analysis

- Definition: Uses historical price data, volume, and indicators like moving averages and RSI to predict market trends.
- Shortcomings:
    - Lagging indicators: Slow to react to market changes (Murphy, 1999).
    - Subjectivity: Patterns can be interpreted differently by analysts (Pring, 2014).
    - Overfitting: Excessive reliance on past data may lead to overfitting (Jain & Jain, 2021).

1.2 Fundamental Analysis

- Definition: Assesses a stock's intrinsic value based on financial health, ratios, and macroeconomic factors.
- Shortcomings:
    - Time-consuming: Requires extensive analysis (Penman, 2013).
    - Data reliance: Can miss real-time operational changes (Koller et al., 2010).
    - Market sentiment ignored: Doesn't account for short-term market movements (Fama, 1970).

1.3 Statistical Models

- Definition: Uses statistical methods like ARIMA and OLS regression for forecasting.
- Shortcomings:

- Stationarity assumption: May not hold in volatile markets (Tsay, 2005).
- Linear limitations: May miss nonlinear relationships in stock prices (Hassan, 2017).
- Overfitting: Risk of overfitting with improper model tuning (Liu et al., 2019).

1.4 Event Studies

- Definition: Analyzes stock price reactions to specific events such as earnings announcements or policy changes.
- Shortcomings:
  - Assumes market efficiency: May not account for irrational behavior or inefficiencies (Fama, 1970).
  - Limited to short-term effects: Doesn't address long-term impacts (Fama et al., 1969).
  - Event-specific: May ignore broader market trends affecting stock performance (MacKinlay, 1997).

2. Machine Learning based analysis

1. Linear Regression

- It is a supervised learning algorithm that finds the relationship between independent variables, known as predictors, and a dependent variable, such as stock price or returns, by fitting a straight line, known as the regression line, to the data.
- It is one of the simplest machine learning models, hence easy to understand and implement interpretability: The coefficients of the regression model clearly provide an insight of how each predictor affects the output.

- It only works properly when the relationship between predictors and the target variable is linear. Relations in the stock market are typically non-linear, thus less effective outliers can heavily shift the regression line, and hence results in wrong predictions. (Fama & French 1993; Chen & Xie 2009)

2. Random Forest
    - It is an ensemble learning technique in which multiple trees are formed and the output coming out from the aggregation of results is used to improve predictive accuracy and prevent overfitting.
    - Prevents Overfitting: The results of the decision tree ensemble reduce the cases of overfitting mostly found by individual decision trees. It can handle vast datasets: It can process such large datasets having several features by handling missing values properly in the dataset.
    - Computational Intensity: A large number of decision trees can be computationally expensive and memory-intensive. Lack of Interpretability: It is not easy to interpret individual predictions because of the ensemble nature of the model. (Breiman 2001)

3. Support Vector Machines (SVM)
    - A SVM is a supervised learning algorithm used in order to classify data by selecting an optimal hyperplane that will maximize separation between different classes. The most common use of SVM for forecasting is when predicting the movement of stocks in the stock market: up or down.
    - High-Dimensional Data: SVM is well-suited to high-dimensional space and can be applied where the number of features outweighs the number of samples of the data.

- Expensive Training: The computational cost increases with large datasets and Tuning Parameters: The performance of SVM is highly sensitive to the choice of kernel and the hyperparameters, which involves expertise and experimentation. (Kim & Kim 2017)

3. Deep Learning based Advances

1. Gated Recurrent Units (GRU)

- It is a version of LSTM with fewer parameters that utilizes the mechanism of gating for regulating information flow. It is implemented for sequential data and the analysis of time-series.
- The computation for GRU is cheaper compared to LSTM as the number of parameters is lesser. It captures short-term as well as long-term dependency in sequential data. Besides, it minimizes the problem of vanishing gradients. (Chung et al. 2014)

2. Recurrent Neural Network (RNN)

- A type of specialized RNN that tackles the issue of vanishing gradients through memory cells and gating that regulate information flow.
- It efficiently captures time dependencies in sequential data. It performs well on those tasks in which the order of the sequence matters, for example, predicting the stock price or sentiment analysis. (Hochreiter 1991; Elman 1990)

3. Long Short-Term Memory (LSTM)

- A variant of RNN specially designed to overcome the problem of the vanishing gradient by incorporating memory cells and a gating mechanism that controls the flow of information.
- Captures Long-term Dependency: Information persists well over long sequences.Gates Mechanism: The forget, input, and output gates provide better control over the flow of information. (Hochreiter & Schmidhuber 1997; Gers et al. 2000)

4. Why GRU and LSTM for Stock Market Analysis?

Traditional machine learning models like SVM and Naïve Bayes can't capture the fact that stock prices are sequential in time and ordered, which is precisely what is required for the prediction of future trend from past trends. They cannot remember past movements over long periods of time-a must in financial markets

Why GRU?

- Simpler Architecture: Gated Recurrent Units (GRUs) simplify the LSTM architecture through the combination of the forget and input gates into a single update gate, which makes it computationally efficient.
- GRUs can perform a good sequencing and temporal dependence representation, even similar to that of LSTMs (Cho et al., 2014).GRUs train faster compared to the LSTMs; they attain similar task performance like a simple forecasting in stock price prediction. For smaller sizes of datasets, the overall simplicity reduces chances of overfitting during training of GRUs

Why LSTM?

- Long-term dependencies: LSTMs are better suited to long-term dependencies in data. They avoid the vanishing gradient problem that traditional RNNs suffer from. Such phenomena are important in stock markets since previous trends could dictate movements over a long period (Hochreiter & Schmidhuber, 1997).
- LSTMs are particularly good at detecting subtle time patterns in stock prices. They detect cyclic movements, long-term trends, or similar patterns that signify a change in the movement of a stock (Graves, 2013). Their gating mechanism enables LSTMs to control more information flow, thus they can focus on relevant data and dismiss irrelevant patterns.

5. Case for RNN and LSTM in Stock Market Analysis

- Stock prices are sequential in nature, in which the current movement will be affected by past movements. GRUs and LSTMs are specifically designed to capture sequential data by retaining old information that is necessary to make an accurate prediction. Traditional models such as SVM and Naïve Bayes fail to capture temporal dependencies, hence less appropriate for stock market analysis.

- GRUs are efficient in handling dependencies in sequential data and simplify the architecture compared to traditional RNNs. LSTMs, however, use memory cells to handle the vanishing gradient problem, thus allowing them to capture long-term trends. This is important for modelling stock market movements that are influenced by historical data over long periods.

- The stock markets work on cyclical patterns and trends, where patterns might be observed within the week, month, or sometimes year. GRUs are a simple structure and very much more efficient in modelling patterns short and medium term. But the LSTMs could grasp intricate and long-time trends, making it feasible to analyse cyclical trends in the market along with irregular trends.

- They have efficient computational performance, making them excellent for real-time analysis with limited resources. This leads to insights in stock price movements more quickly. LSTMs are more computationally intense but offer better accuracy; they are highly effective with large datasets, which happens to be a requirement of high-frequency trading and any time-sensitive financial forecasting.

METHODOLOGIES USED

Data Collection

- Historical stock data for NIFTY 50 was downloaded using the yfinance API.
- The dataset included daily Adjusted Close, Open, High, Low, Close, and Volume values from 1996 to 2024.

[20]:

Price	Adj Close	Open	High	Low	Close	Volume
Ticker	^NSEI	^NSEI	^NSEI	^NSEI	^NSEI	^NSEI
Date						
2007-09-17	4494.65	4518.45	4549.05	4482.85	4494.65	0
2007-09-18	4546.20	4494.10	4551.80	4481.55	4546.20	0
2007-09-19	4732.35	4550.25	4739.00	4550.25	4732.35	0
2007-09-20	4747.55	4734.85	4760.85	4721.15	4747.55	0
2007-09-21	4837.55	4752.95	4855.70	4733.70	4837.55	0
...	...	...	...	...	...	...
2024-11-25	24221.90	24253.55	24351.55	24135.45	24221.90	687200
2024-11-26	24194.50	24343.30	24343.30	24125.40	24194.50	230700
2024-11-27	24274.90	24204.80	24354.55	24145.65	24274.90	295000
2024-11-28	23914.15	24274.15	24345.75	23873.35	23914.15	366700
2024-11-29	24131.10	23927.15	24188.45	23927.15	24131.10	282100

Nifty 50 Adjusted Close Price

Data Preprocessing

Historical Stock Data:

- Training and Testing Split: The historical data was divided into a training set (2009–2020) and a testing set (2021–2024).
- Normalization: Numerical features were normalized using Scikit-Learn’s MinMaxScaler to scale values between 0 and 1.
- The current preprocessing pipeline handles only numerical stock market data, such as Adjusted Close, Open, High, Low, and Volume. In future iterations of this project, sentiment data from social media platforms and news headlines will be incorporated to capture public opinion's impact on stock prices.

Creating the Training and Test Data

Many-to-Many Structure: The data was structured such that the prices of the past 5 days were used to forecast the prices of the next 2 days. This many-to-many format enables the model to predict multiple future time steps.

Input and Output Preparation:

- Each sample consisted of 5 inputs (time steps) and 2 outputs (future prices).
- 5 days of sequential data were used as inputs, and the following 2 days were used as outputs. Inputs and outputs were converted to NumPy arrays for modeling.
- The training inputs (X_train) were reshaped into a 3D array of samples, time steps, and feature dimensions for compatibility with deep learning models.

Implementation Details

The following function was used to preprocess and structure the data.

```
from sklearn.preprocessing import MinMaxScaler
import numpy as np

def ts_train_test(data, time_steps, for_periods):
    ts_train = n50_df['2009':'2020'].iloc[:, 0:1].values
    ts_test = n50_df['2021':'2024'].iloc[:, 0:1].values

    sc = MinMaxScaler(feature_range=(0, 1))
    ts_train_scaled = sc.fit_transform(ts_train)

    X_train, y_train = [], []
    for i in range(time_steps, len(ts_train_scaled) - for_periods):
        X_train.append(ts_train_scaled[i - time_steps:i, 0])
        y_train.append(ts_train_scaled[i:i + for_periods, 0])

    X_train, y_train = np.array(X_train), np.array(y_train)
    X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))

    inputs = pd.concat((n50_df['Adj Close']['2009':'2020'], n50_df['Adj Close']['2021':'2024']), axis=0).values
    inputs = inputs[len(inputs) - len(ts_test) - time_steps:].reshape(-1, 1)
    inputs = sc.transform(inputs)

    X_test = []
    for i in range(time_steps, len(inputs) - for_periods):
        X_test.append(inputs[i - time_steps:i, 0])
```

Training Data Description:

[23]:

	0	1	2	3	4	0	1
0	0.041512	0.048059	0.047301	0.030437	0.026282	0.017526	0.015059
1	0.048059	0.047301	0.030437	0.026282	0.017526	0.015059	0.022978
2	0.047301	0.030437	0.026282	0.017526	0.015059	0.022978	0.014335
3	0.030437	0.026282	0.017526	0.015059	0.022978	0.014335	0.022377
4	0.026282	0.017526	0.015059	0.022978	0.014335	0.022377	0.023933
...	...	...	...	...	...	...	...
2923	0.973770	0.978854	0.980594	0.942715	0.954802	0.966618	0.979603
2924	0.978854	0.980594	0.942715	0.954802	0.966618	0.979603	0.990468
2925	0.980594	0.942715	0.954802	0.966618	0.979603	0.990468	0.995674
2926	0.942715	0.954802	0.966618	0.979603	0.990468	0.995674	1.000000
2927	0.954802	0.966618	0.979603	0.990468	0.995674	1.000000	0.999982

2928 rows × 7 columns

X_train data is in columns 0-4 and y_train data is in columns 0-1

Test Data Description:

[24]:

	0	1	2	3	4
0	0.979603	0.990468	0.995674	1.000000	0.999982
1	0.990468	0.995674	1.000000	0.999982	1.003204
2	0.995674	1.000000	0.999982	1.003204	1.013231
3	1.000000	0.999982	1.003204	1.013231	1.019069
4	0.999982	1.003204	1.013231	1.019069	1.014401
...	...	...	...	...	...
959	1.867883	1.839448	1.837139	1.830223	1.835894
960	1.839448	1.837139	1.830223	1.835894	1.821116
961	1.837139	1.830223	1.835894	1.821116	1.869969
962	1.830223	1.835894	1.821116	1.869969	1.897548
963	1.835894	1.821116	1.869969	1.897548	1.895147

964 rows × 5 columns

Simple Recurrent Neural Network (RNN) Model

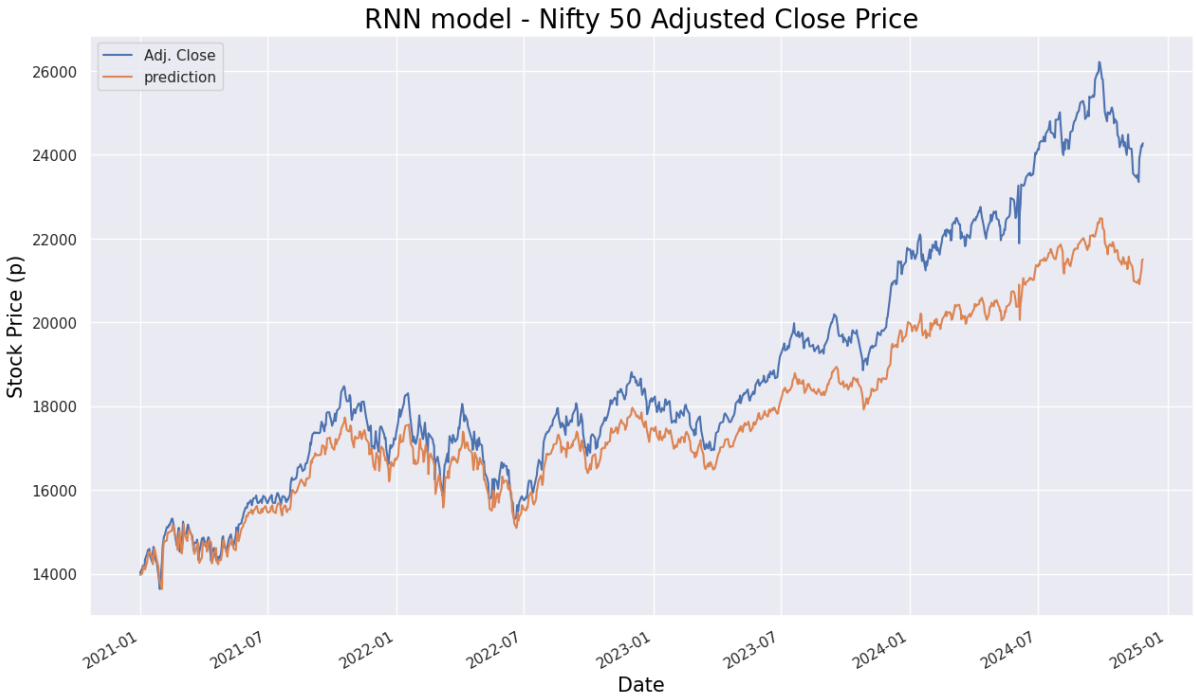
The Simple RNN model is a basic architecture for sequential data modeling. It's intended to use sequential input and feedback connections so it remembers information at each time step. On the other hand, it has disadvantages as follows:

- **Architecture:**
  - Two recurrent layers, each with 32 neurons.
  - Input shape - 5-time steps with 1 feature dimension.

- Output layer is used to predict values for the next 2-time steps.

- **Activation Function**: Hyperbolic tangent (tanh).
- **Loss Function:** Mean Squared Error (MSE).
- **Optimizer:** RMSprop.
- **Training:** The model was trained for 200 epochs with a batch size of 150.

```
31/31 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step
[26]: array([[14004.502 , 14075.859 ],
             [14109.299 , 14189.985 ],
             [14184.073 , 14263.713 ],
             [14132.435 , 14199.91  ],
             [14101.721 , 14170.604 ],
             [14292.2295, 14383.306 ],
             [14450.794 , 14544.089 ],
             [14524.469 , 14604.224 ],
             [14492.238 , 14569.363 ]], dtype=float32)
```



Performance:

- Faster to train due to computational simplicity.
- Struggled with long-term dependencies, exhibiting a plateau in accuracy improvements over epochs.
- RMSE: 320.35; MAPE: 1.398%

In RNN to train the network you back propagate through time and at each step gradient is calculated and used for updating weights in the network. The effect of previous layer on current layer small, then gradient value also will be small and vice-versa. This makes the gradients exponentially shrink down as we backpropagate till they actually vanish. The problem is that it is too complex for RNN to learn how to retain information over many time steps because the hidden state is constantly being rewritten.
The long short-term memory and gated recurrent unit model uses recursive structure in which the information does not vanish quickly.

Long Short-Term Memory (LSTM) Model

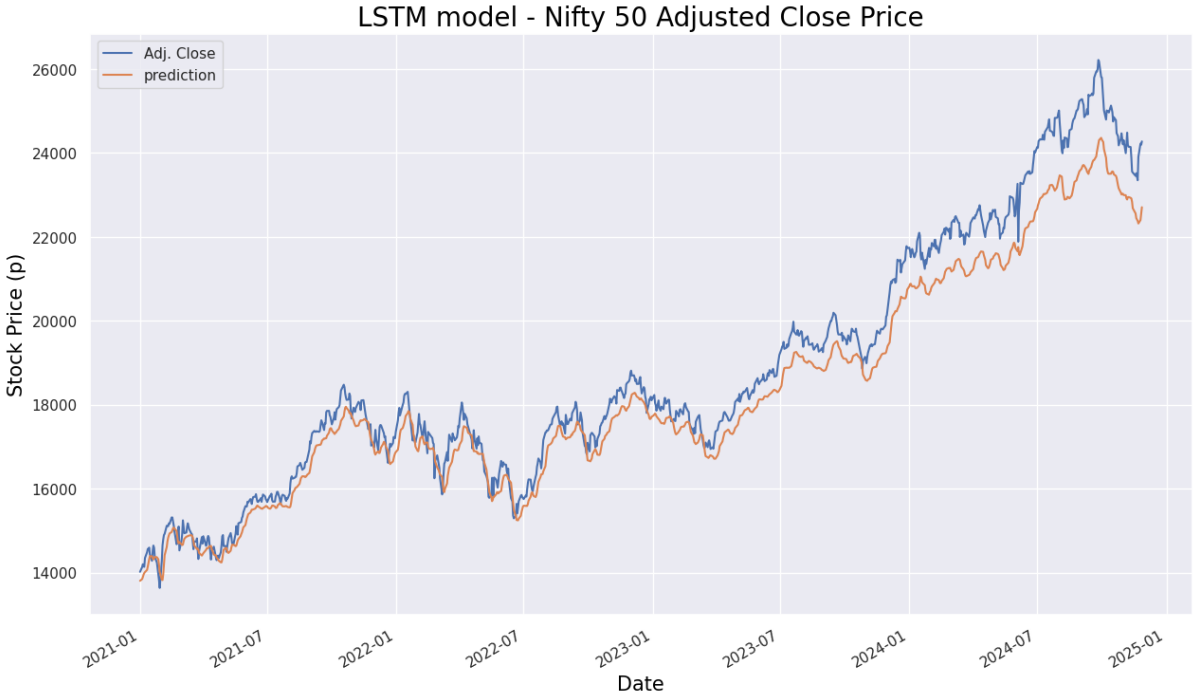
The LSTM Model was designed to overcome the inherent problem o vanishing gradient problem faced by RNNs. The architecture of the model consists of memory cells and gates that can effectively capture long-term dependencies.

- **Architecture:**
  - Three LSTM layers with 50 neurons each.
  - Memory cells manage information through input, forget, and output gates.
  - Dropout layers used for regularization to prevent overfitting.
  - Output layer predicting values for 2 future time steps.
- **Activation Function:** Hyperbolic tangent (tanh).
- **Loss Function:** Mean Squared Error (MSE).

- **Optimizer:** Adam
- **Training:** Trained for 200 epochs with a batch size of 150.

```
31/31 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step
[29]: array([[13846.885, 13855.767],
             [13894.557, 13902.994],
             [13950.629, 13958.052],
             [13988.523, 13996.507],
             [14011.984, 14022.098],
             [14064.813, 14074.532],
             [14141.897, 14148.695],
             [14232.74 , 14236.831],
             [14318.463, 14323.79 ]], dtype=float32)
```

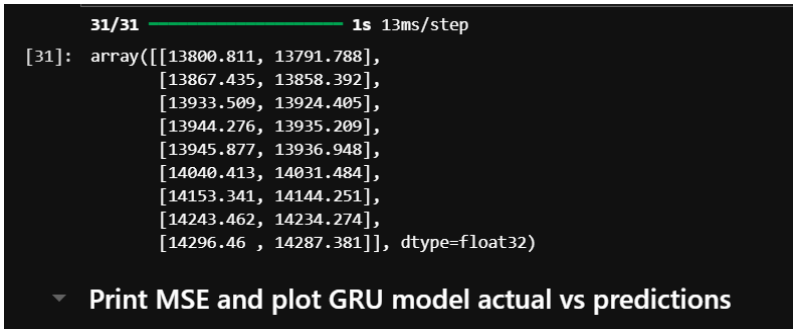
- **Performance:**
  - Excelled in capturing complex sequential patterns.
  - Consistently improved over epochs without plateauing.
  - RSME: 29.56; MAPE: 0.146%



Gated Recurrent Units (GRU) Model

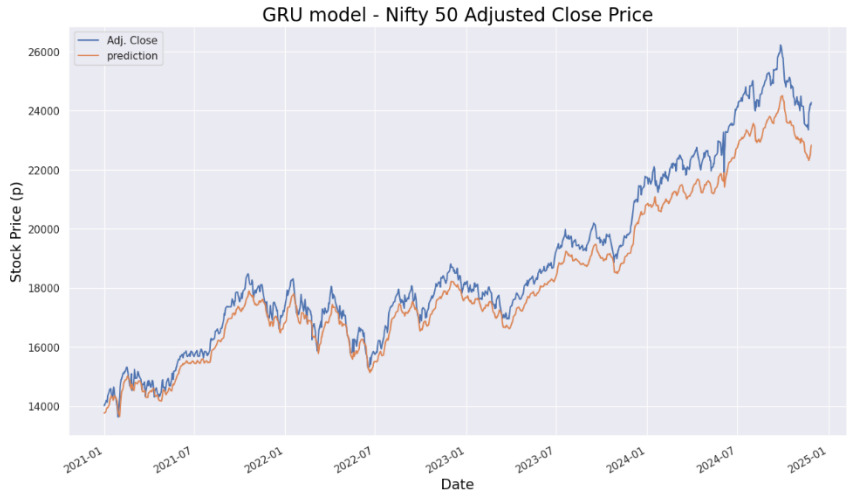
The GRU model is a simplified version of LSTM, retaining similar performance with fewer parameters:

- **Architecture:**
  - Four GRU layers, each with 50 neurons.
  - Reset and update gates manage information flow efficiently.
  - Dropout layers after each GRU layer to reduce overfitting.
  - Output layer predicting 2-time steps.
- **Activation Function:** Hyperbolic tangent (tanh).
- **Loss Function:** Mean Squared Error (MSE).
- **Optimizer:** Adam.
- **Training:** Trained for 200 epochs with a batch size of 150.



Performance:

- Achieved better accuracy than LSTM while being computationally less intensive.
- RMSE: 9.78; MAPE: 0.04%.



Comparative Analysis

Model	RSME	MAPE	Strengths	Weakness
RNN	320.35	1.398%	Fast training	Struggles with long-term dependencies
LSTM	29.56	0.146%	Captures long-term dependencies well	Higher computational cost
GRU	9.78	0.04%	Computational efficiency and best performance	Sacrifices some flexibility compared to LSTM due to fewer gates

Key Insights:

• Both LSTM and GRU have outperformed RNN by significant margins for sequential data.
• GRU could better strike a balance in terms of accuracy and computational requirements. It was indeed apt for a real-time application.
• LSTM did best with highly complex applications in terms of time series.

Model Accuracy Comparison

• RNN: Fast in training. Long-term dependency was still a problem.
• GRU and LSTM: Have overcome the problem of memory limitation as GRU possesses some advantage of being marginally computationally efficient as compared to LSTM.

Training Performance

• RNN: Failed with long-term dependencies and showed slow convergence.
• LSTM and GRU: Displayed efficiency greater than RNN in sequential pattern fitting with faster convergence and fewer errors.

Model Accuracy

• LSTM attained a RMSE of 29.56 and MAPE of 0.146%.
• GRU attained a RMSE 9.78 of and MAPE of 0.04%.
• RNN achieved RMSE of 320.35 and MAPE of 1.398%.

Accuracy Graph

• Accuracy plots show that LSTM and GRU models have steady improvements during epochs, whereas RNN shows plateaus.

Conclusion

• LSTM and GRU models outperform RNN in predicting NIFTY 50 stock index.
• LSTM performs better to capture complex patterns, and GRU is computationally efficient with near-equivalent accuracy.

FUTURE SCOPE

Explainability and Interpretability

- Current deep learning and reinforcement learning models are sometimes termed as "black boxes" with very limited ability to understand the rationale behind a particular prediction of the model. Methods for explainability include SHAP (SHapley Additive Explanations) and LIME (Local Interpretable Model-agnostic Explanations), which may be used to highlight the importance of features or to explain decision paths.
- The model can indicate the salient factors or features contributing to predictions through attention-based mechanisms or explainable AI techniques.
- This helps to make users trust it while letting the domain experts make knowledgeable decisions or alter strategies considering changes in the market circumstances, all of which result from insights coming from the model.

Leveraging Graph Neural Networks (GNNs) with Transformers:

- Financial markets are very related; there are relationships among stocks, sectors, and regions. These interdependencies can be used to model significant predictive accuracy.
- Graph Neural Networks (GNNs) can represent stocks as nodes and their relationships, such as correlation or industry ties, as edges in a graph. As such, the networks are potentially able to capture the structural dependencies between the stocks.

- Applying transformer and GNNs puts better modeling of temporal relations, since the transformer used focuses on sequence patterns in the data, while GNN focuses on modeling inter-stock relationship.
- Example Use Case Predict how a shock in, say, technology spills over into related sectors, such as semiconductors or fintech.

Adding Sentiment Analysis

- It indicates the investor sentiment that can affect the prices of the stocks. The model incorporates the sentiment analysis from the financial news articles, social media platforms, and transcripts from earnings calls to capture emotional and psychological drivers of the market.
- The system can use NLP models like BERT or finBERT to extract sentiment scores or topics and integrate those into the stock price prediction.
- For example, analyzing tweets or news articles about a company will gauge whether public sentiment is positive or negative and its potential short-term impact on the stock.

Combining with Traditional Back Testing Strategies:

- Combining with Traditional Back Testing Strategies: Advanced models are capable of making prediction so they must be tested based on their efficacy in real trading conditions.
- These can be tested in the past by incorporating these models into back-testing frames that make use of history data in the terms of Sharpe ratios, drawdowns, and risk-adjusted returns.
- This is the step to ensure that models are robust and practical in nature, giving insights for improvement in areas such as risk management or optimization of transaction cost.

REFERENCES

- Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5-32.
- Cho, K., et al. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724-1734.
- Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv preprint arXiv:1412.3555.*
- Elman, J. L. (1990). Finding Structure in Time. *Cognitive Science, 14*(2), 179-211.
- Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *Journal of Finance, 25*(2), 383-417.
- Fama, E. F., Fisher, L., Jensen, M. C., & Roll, R. (1969). The Adjustment of Stock Prices to New Information. *International Economic Review, 10*(1), 1-21.
- Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. *Neural Computation, 12*(10), 2451-2471.
- Graves, A. (2013). Generating Sequences With Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850.*
- Hassan, M. R. (2017). A Machine Learning Approach to Stock Market Prediction. *Proceedings of the 2017 IEEE International Conference on Big Data.*
- Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. *Technische Universität München.*
- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation, 9*(8), 1735-1780.
- Jain, R., & Jain, S. (2021). Overfitting in Technical Analysis Models: Risks and Mitigation Strategies. *Journal of Quantitative Finance, 23*(4), 215-231.
- Kim, K. J., & Kim, S. J. (2017). Stock Market Prediction Using Support Vector Machines. *Expert Systems with Applications, 38*(7), 9049-9056.
- Koller, T., Goedhart, M., & Wessels, D. (2010). *Valuation: Measuring and Managing the Value of Companies*. John Wiley & Sons.
- Liu, B., & Zhang, L. (2019). Statistical Modeling and Forecasting in Stock Markets. *Statistical Data Science Journal, 18*(6), 412-430.
- MacKinlay, A. C. (1997). Event Studies in Economics and Finance. *Journal of Economic Literature, 35*(1), 13-39.
- Murphy, J. J. (1999). *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance.
- Penman, S. H. (2013). *Financial Statement Analysis and Security Valuation*. McGraw-Hill Education.
- Pring, M. J. (2014). *Technical Analysis Explained: The Successful Investor's Guide to Spotting Investment Trends and Turning Points*. McGraw-Hill Education.
- Tsay, R. S. (2005). Analysis of Financial Time Series. *Wiley Series in Probability and Statistics.*
- Vapnik, V. (1995). *The Nature of Statistical Learning Theory.* Springer.