Topic : Option Pricing Using Machine Learning

Presented by:
Ayushman Tamrakar – 2240401123
Semester – 6th

Minor Project
Sub Code – MDS 328

Mathematics, Bioinformatics and Computer Applications

Introduction

- **Options**

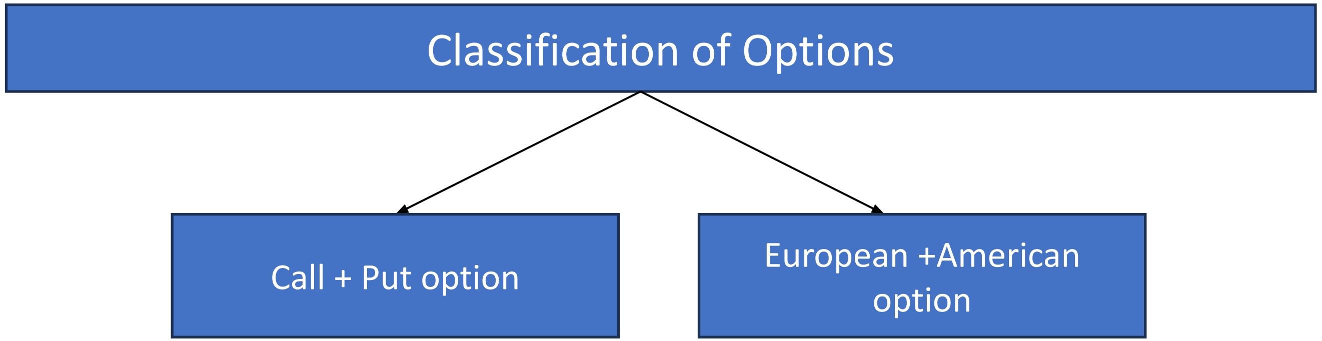
Option pricing refers to the process of determining the fair value or theoretical price of an options contract. An option is a financial derivative that gives the buyer the right, but not the obligation, to buy or sell an underlying asset (like a stock) at a predetermined price (strike price) before or at the expiration date.

- **Owner of the option**

Owner/Buyer/Holder of the option is the party who has the leverage to either buy/sell the underlying stock.

- **Seller of the option**

Seller/Writer of the option is the party which is under obligation and the obligation is to sell/buy the underlying stock.



- **Call option**

The option where the owner has the right to buy the underlying.

- **Put option**

The option where the owner has the right to sell the underlying.

- **European option**

The option where the owner can buy/sell the underlying asset on a fixed/given future date.

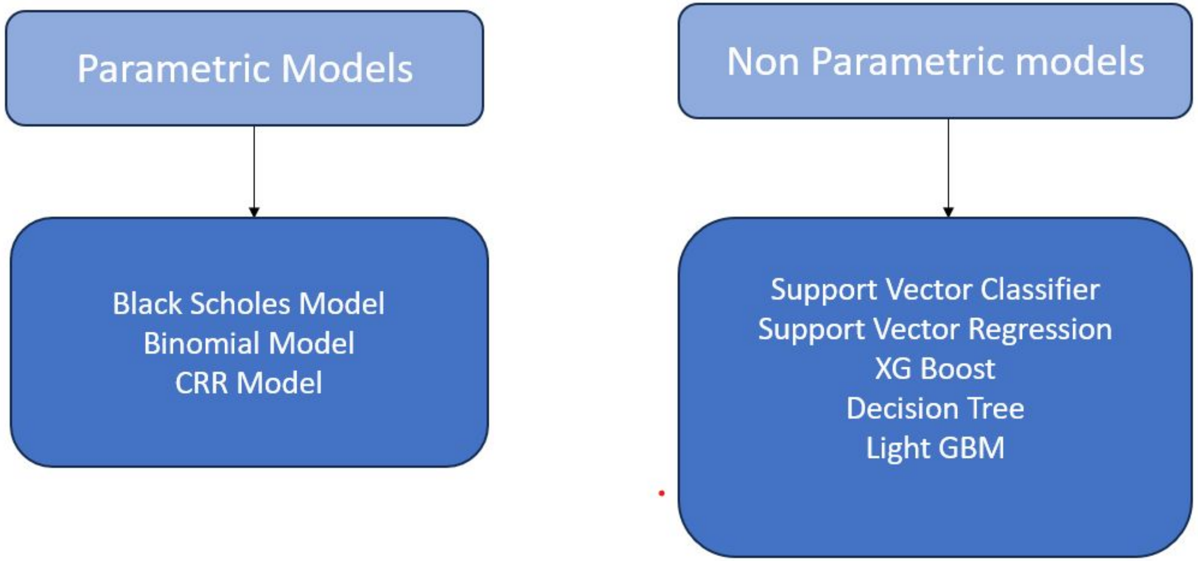
- **American option**

The option where the owner can buy/sell the underlying asset up to (and including) fixed/given future date.

Challenges in Predicting Option Prices

- **High Volatility:** Option prices are extremely sensitive to changes in market conditions and underlying asset prices.
- **Multifactor Dependence:** Option pricing is influenced by multiple variables, including the price of the underlying asset, time to expiration, interest rates, and implied volatility.
- **Non-linear Relationships:** The relationship between input factors and the option price is complex and often non-linear.
- **Market Sentiment & Events:** Unpredictable factors such as news, earnings reports, or macroeconomic changes can heavily impact prices.
- **Lack of Clean Data:** Noise and inconsistencies in financial data can hinder model performance.

Existing Approaches or Models in Option Pricing



Dataset Overview

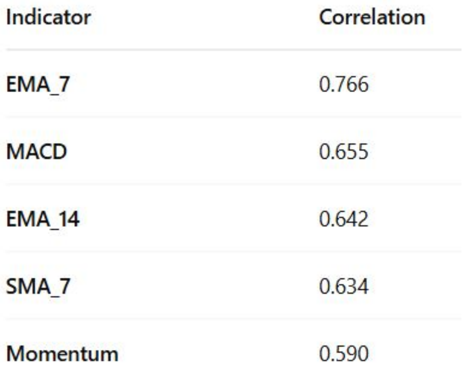
- This dataset is a historical data of NIFTY 50 index daily prices data taken from NSE website. All datasets are at a day-level with pricing and trading values split across.
  The datasets contains the below columns:

Trade Date - The trading day on which the data was recorded.
Expiry Date – The date on which the option contract expires.
Strike Price - The pre-agreed price at which the option can be exercised.
Open - Day's open price
High - Day's high price
Low - Day's low price
Close - Day's close price
Volume - Refers to the number of option contracts traded during a particular trading day.

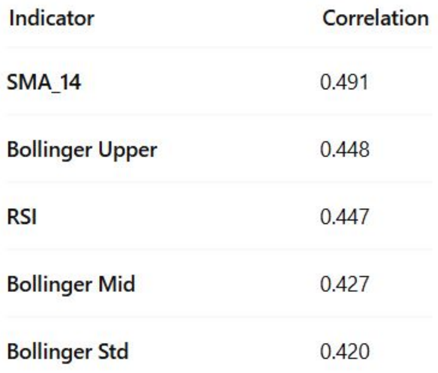
- Data ranges from 01 Mar 2024 to 30 Aug 2024 in the dataset which contains 1544 records .
- Open, high, low are taken as features and close is taken as target variable. 80% data is taken as training data and 20% of the data as testing data.

Technical Indicators and Feature Engineering

Technical indicators are mathematical calculations based on historical price, volume, or open interest data. They help traders and models **identify trends, momentum, volatility, and potential reversal points** in financial markets. These indicators translate complex price actions into quantitative features that a machine learning model can interpret. They help in understanding **price direction**, **strength of movement**, and **market noise**, all of which directly affect an option's close price.
These features were selected based on (**statistical correlation > 0.60** with the Close Price) and their financial interpretability.



Indicator	Correlation
EMA_7	0.766
MACD	0.655
EMA_14	0.642
SMA_7	0.634
Momentum	0.590



Indicator	Correlation
SMA_14	0.491
Bollinger Upper	0.448
RSI	0.447
Bollinger Mid	0.427
Bollinger Std	0.420

In **Feature Engineering** a range of technical indicators using OHLCV data and expiry timelines.
Added days to expiry and momentum as custom features.
Removed noise using rolling averages and exponential smoothing.

Models Applied and Their Errors

We have applied total six models for the prediction of Close price. The followings are the outputs we got and we have also plotted the graph for each model and in last did the comparison all the applied models.

Models	**Root Mean Square Error RMSE**	**Mean Absolute Error MAE**	R^2 Score
XG Boost (m1)	6.4235	8.9564	0.9992
Neural Network (m2)	16.4972	22.3714	0.9947
Weighted Ensembled (m1+m2)	9.2795	11.7127	0.9986
Stacked Ensembled (m1+m2)	5.9647	7.8970	0.9993
AM + MLP	17.5412	23.690	0.9943
CNN + Bi-LSTM	19.8540	24.9336	0.9914
Early Binding	**5.878**	**7.708**	**0.9995**
Late Binding	80.826	103.457	0.8899

Final Results:

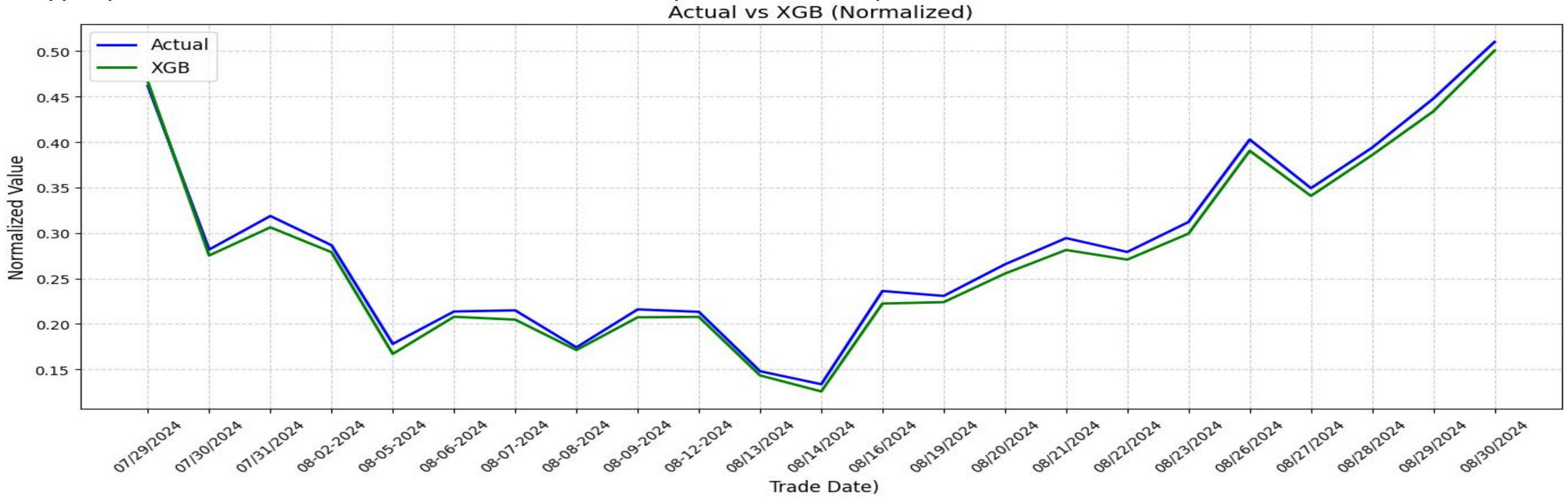
Root Mean Square Error(RMSE) : 5.878

Mean Absolute Error(MAE) :7.708

R^2 Score :0.9995

Optimized XG Boost Model

XG Boost is a powerful ensemble technique based on boosting decision trees. It can efficiently handle large structured datasets and capture complex feature interactions. XG Boost models non-linear dependencies and variable interactions very effectively. It handles missing data, outliers, and noisy financial time series well. It has built-in regularization controls overfitting.
Hyperparameters tuned with GridSearchCV to optimize model performance.

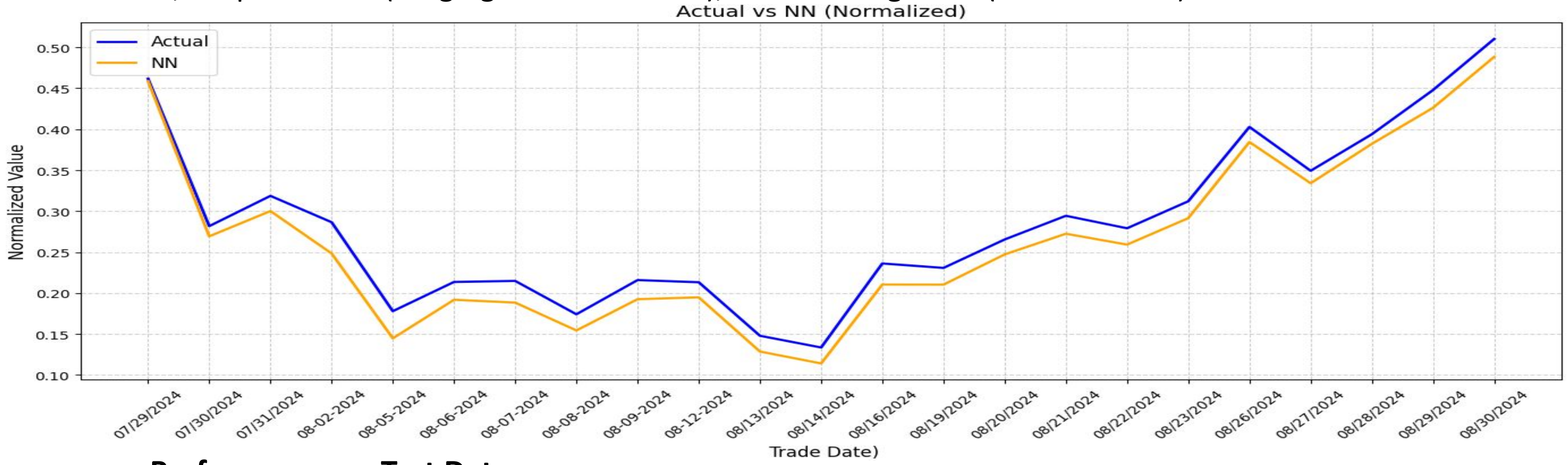


Performance on Test Data:

MAE: ***6.4235*** RMSE: ***8.9564*** R^2 Score: ***0.9992***

Neural Network Model with Keras Tuner

A Fully Connected Neural Network was developed using TensorFlow/Keras to model the complex, non-linear relationships present in financial market data. The architecture consisted of two hidden layers:
The first layer tuned to have between 32 and 96 neurons using **ReLU** activation.
The second layer tuned between 16 and 64 neurons.
A final dense layer with a single output neuron was used for predicting the Close price.
The Keras Tuner with Random Search was employed to find the optimal hyperparameters, including the number of neurons, dropout rates (ranging from 0.2 to 0.5), and learning rates (0.01 or 0.001).



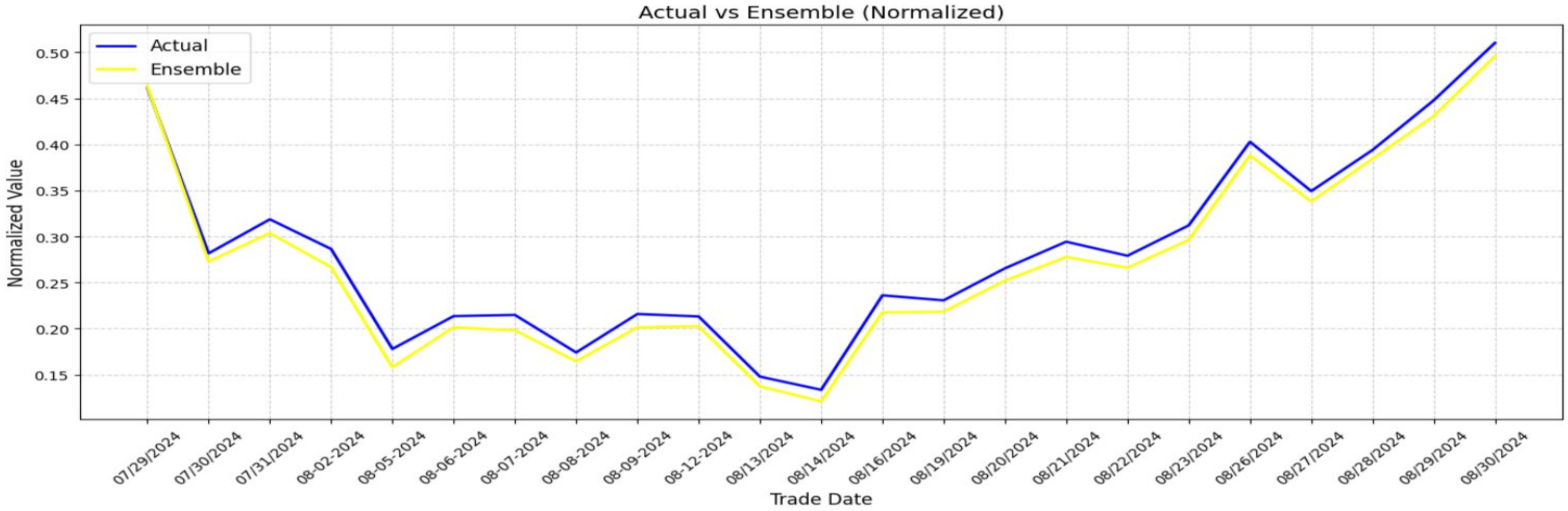
Performance on Test Data:

MAE: ***16.4972*** RMSE: ***22.3714*** R^2 Score: ***0.9947***

Weighted Ensemble Model

An ensemble model combines predictions from multiple models to improve overall performance. In this case, a weighted ensemble of XGBoost and the neural network was created, with weights of **0.6** and **0.4**, respectively.

- **Why Used?** By leveraging the strengths of both models, the ensemble can achieve better generalization and performance on unseen data.



Performance on Test Data:

MAE: *9.2795* RMSE: *11.7127* R^2 Score: *0.9986*

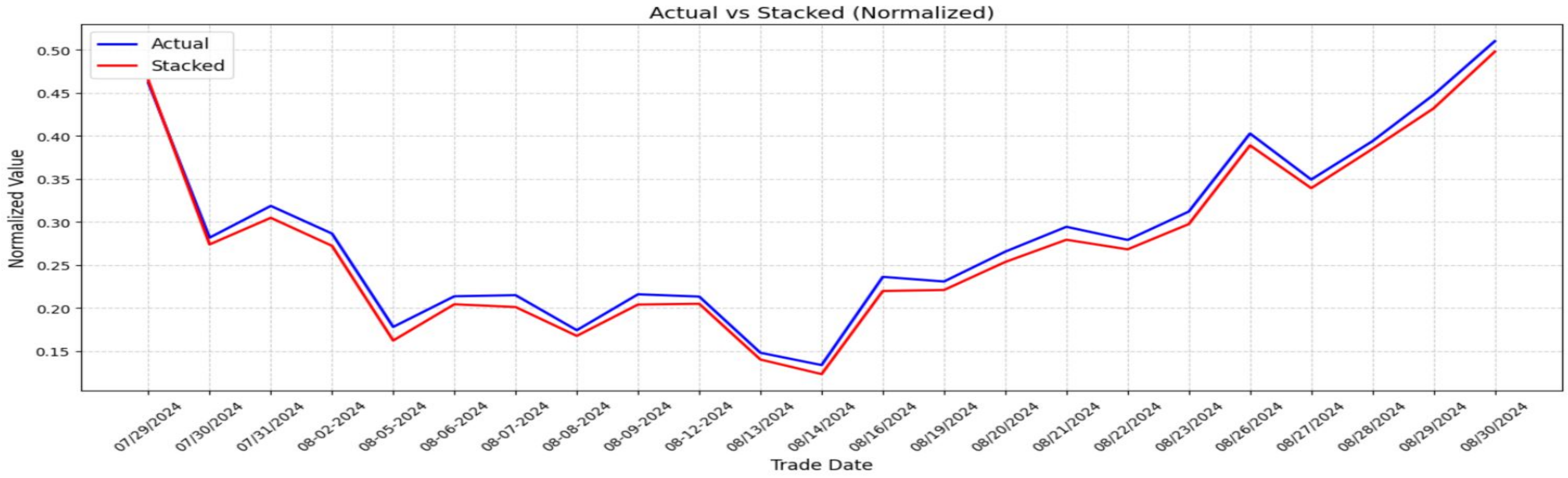
Stacked Model (Ridge Regression)

Stacking is another ensemble technique that combines predictions from base models (XGBoost and NN) as input to a higher-level model. In this case, a **Ridge Regression** model was used as the meta-model.

Why Ridge Regression?

•Helps avoid overfitting by preventing any single model's predictions from dominating.

•Works well for combining predictions that may have collinearity or overlapping information.



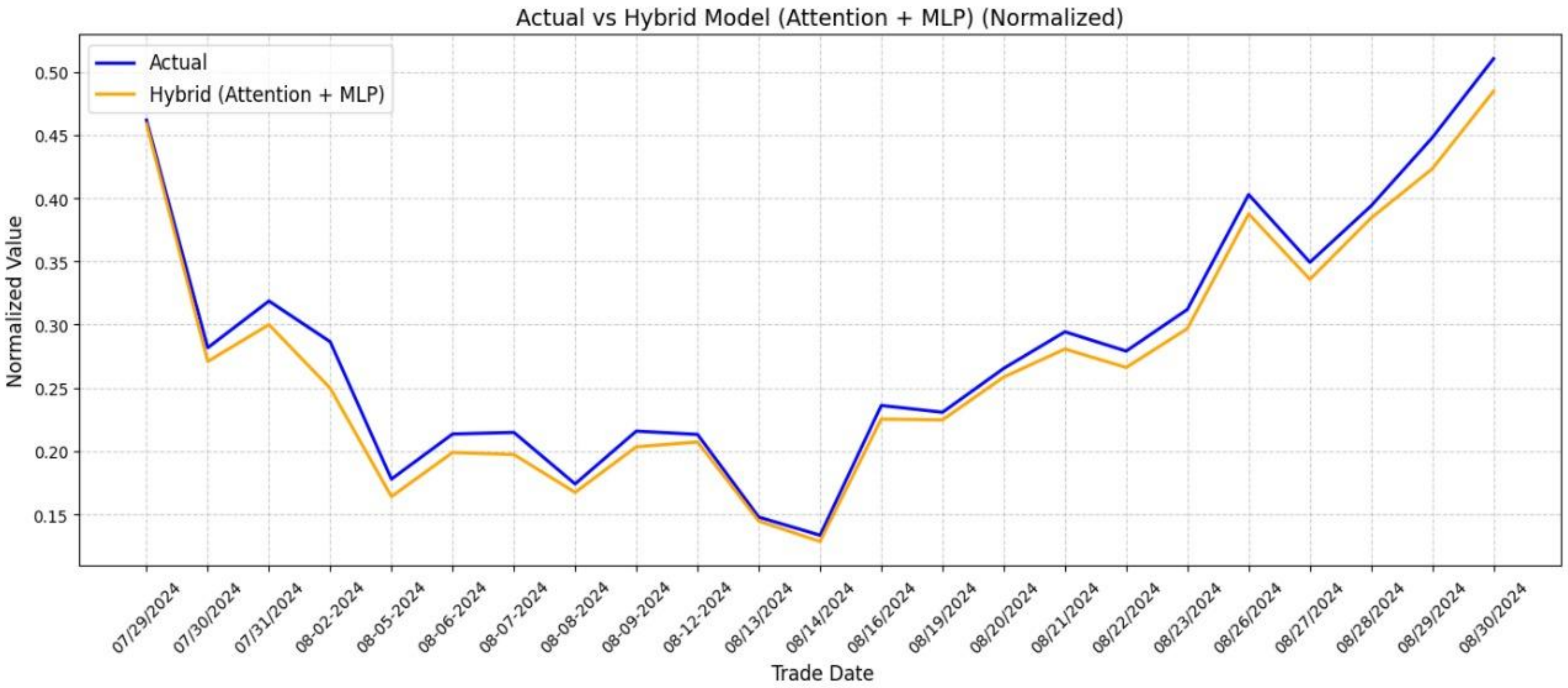
Performance on Test Data:

MAE: 5.9647 RMSE: 7.8970 R^2 Score: *0.9993*

Hybrid Model of (Attention Mechanism + Multi-Layer Perceptron)

- A **Hybrid Neural Network** combines an **Attention mechanism** with a **Multilayer Perceptron (MLP)**.
- It has **two parallel branches**:
  - The **Attention branch** captures important feature interactions and dependencies.
  - The **MLP branch** extracts deep hierarchical patterns from the features.
- The outputs of both branches are **merged early** (early binding) and passed through **additional dense layers** (late binding) for final prediction.
- **Early stopping** is used during training to **avoid overfitting** and improve generalization.
- In **financial time series**, the **Attention mechanism** identifies which technical indicators or market signals are most critical for predicting **Close prices**.
- The **MLP branch** captures **global feature patterns**, learning complicated relationships that traditional models may miss.
- It provides **better model interpretability**, as Attention weights highlight which features influence the prediction most

Graph of Hybrid Model (AM + MLP)



Performance on Test Data:

MAE: ***17.5412*** RMSE: ***23.690*** R^2 Score: ***0.9943***

Hybrid Model of (Convolution Neural Network + Bi-LSTM)

- A Hybrid Neural Network combines CNN and Bi-LSTM branches.
- The CNN branch captures local temporal patterns using convolutional filters.
- The Bi-LSTM branch captures long-term dependencies by processing sequences both forward and backward.
- Outputs of both branches are merged via concatenation, blending local and global information.
- In financial time series, CNN identifies short-term trends, while Bi-LSTM captures long-term market dynamics.
- Early stopping is used to prevent overfitting and improve generalization.
- The late binding strategy helps combine localized features and sequential dependencies effectively.
- In option pricing, the model captures rapid price movements and underlying asset trends more accurately.
- It improves the estimation of option premiums by modeling volatility and time-dependent factors better than traditional Black Scholes-based approaches

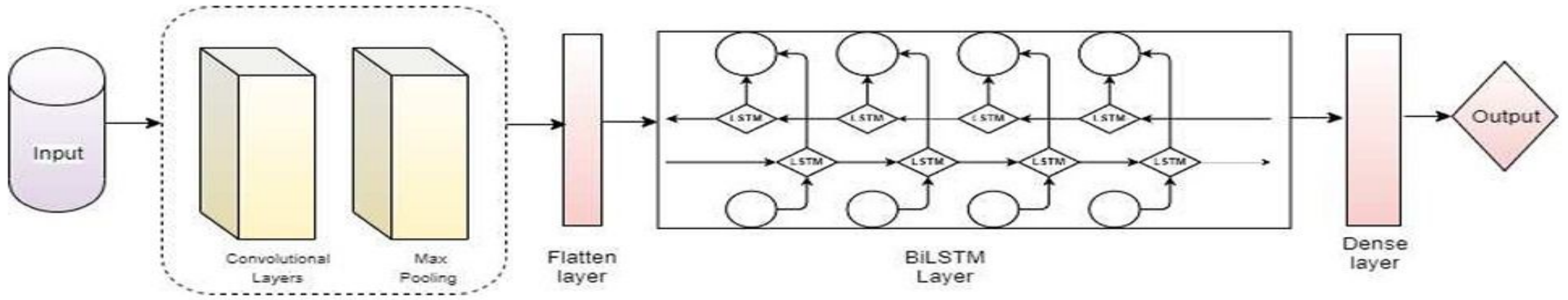
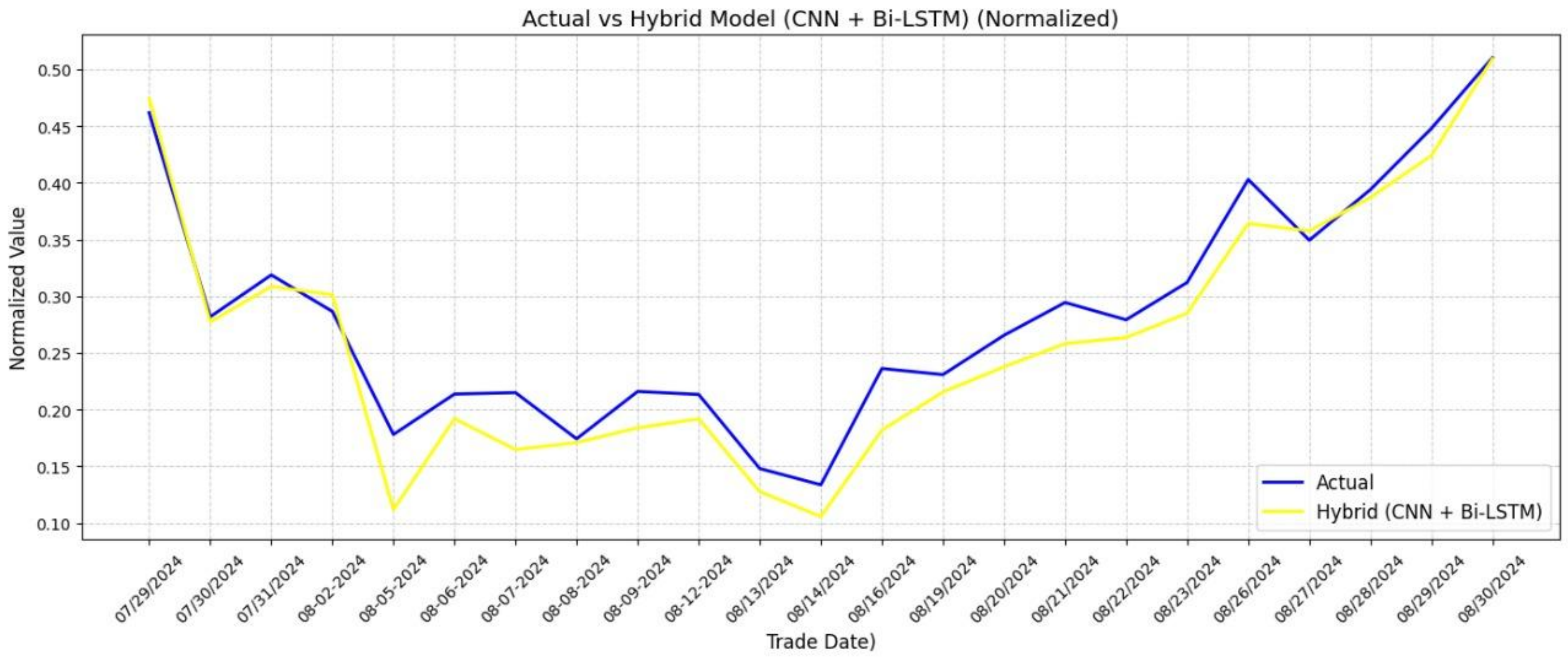


Fig. 2: CNN-BiLSTM

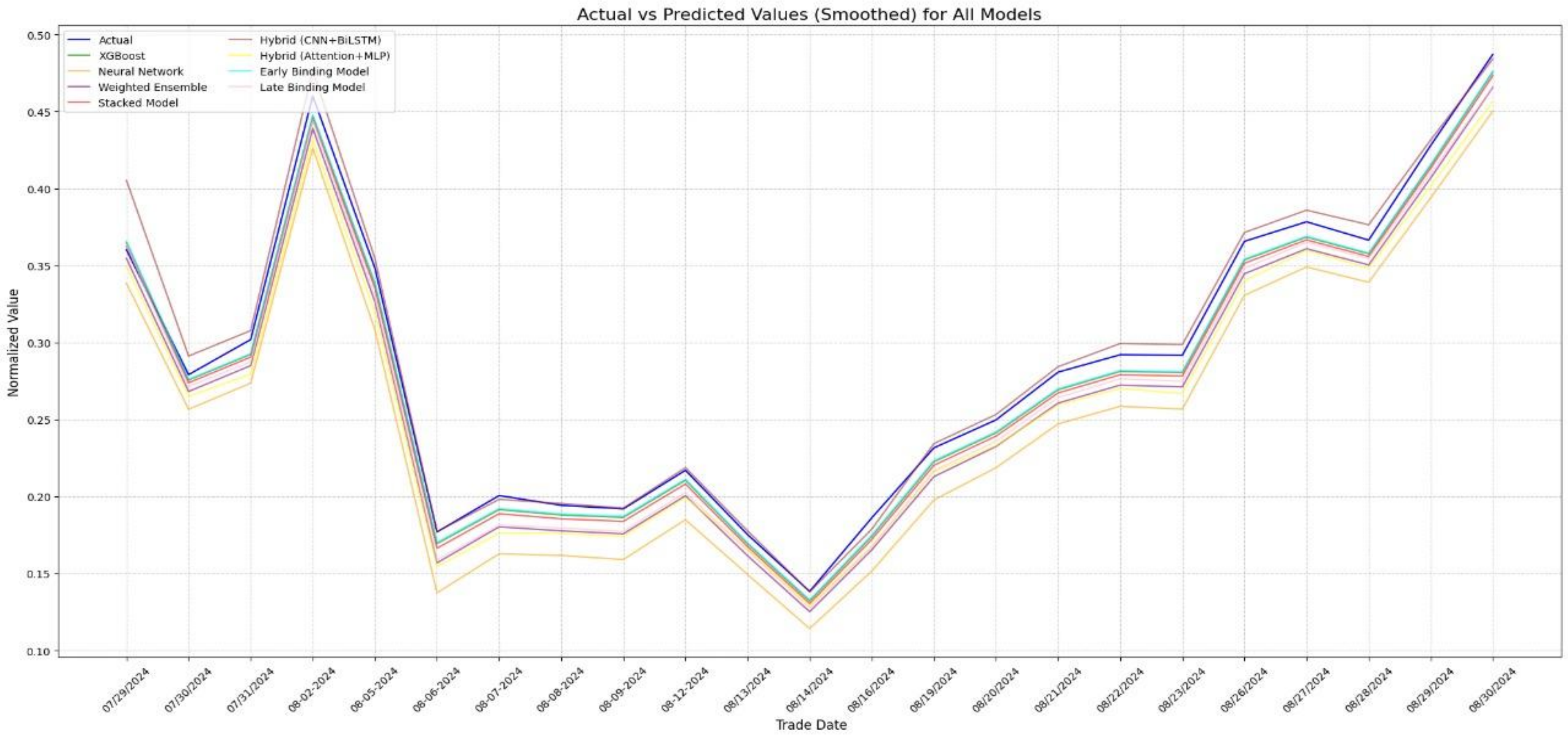
Graph of Hybrid Model (CNN + Bi-LSTM)



Performance on Test Data:

MAE: ***19.8540*** RMSE: ***24.9936*** R² Score: ***0.9914***

Graphical Comparison of all the Applied Models



Conclusion

In this project, various machine learning and deep learning models were explored to predict the Close price of financial options. Models ranged from XG Boost and Neural Networks to advanced hybrids like Attention-MLP and CNN-Bi-LSTM. Feature engineering with technical indicators improved model interpretability and performance. Ensemble methods such as weighted and stacked approaches further enhanced prediction accuracy. The CNN-Bi-LSTM model effectively captured short-term trends and long-term dependencies, while the Attention-MLP model learned intricate feature interactions. Deep learning hybrids using early and late binding strategies demonstrated strong generalization and robust performance, highlighting their potential for real-world financial forecasting applications.

References

- https://www.nseindia.com/
  Source for financial datasets.
- https://www.tensorflow.org/tutorials/images/cnn
  CNN tutorial by TensorFlow.
- https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning/
  Basics of CNNs in ML.
- https://www.geeksforgeeks.org/ml-attention-mechanism/
  Overview of attention mechanism.
- https://www.ibm.com/think/topics/attention-mechanism
  Explanation of attention in ML.
- https://www.geeksforgeeks.org/ml-xgboost-extreme-gradient-boosting/
  Guide to XGBoost algorithm.
- https://www.kaggle.com/
  Datasets and ML competitions.
- https://www.youtube.com/watch?v=huNPPodkFGU

Thank You!