Gradient Boosting Neural Network in American Call Option Pricing

Kamin Atsavasirilert,Ruichen Zhao

2023.12.9

Abstract

This study investigates the application of a Gradient Boosting Neural Network for pricing American call options, utilizing a Heston stochastic volatility model with a Partial Differential Equation (PDE)-based pricing scheme to generate data. The research begins by examining the impact of dataset size on model performance, aiming to identify the optimal dataset size. Subsequently, four models—Decision Tree, Gradient Boosting Tree, Neural Network, and Boosting Neural Network—are fitted, and their prediction Mean Absolute Error (MAE) and processing time are compared. Further, a sensitivity test on the Gradient Boosting Neural Network is conducted to evaluate its strengths and weaknesses, followed by model tuning attempts for performance enhancement.

The experimental results reveal key insights. Firstly, an increase in dataset size contributes to improved model performance, diminishing prediction errors, although the benefits plateau beyond a dataset size of 100 thousand. Secondly, Neural Networks outperform Tree-based models in terms of both accuracy and speed. Additionally, boosting proves beneficial in error reduction across all models. However, the sensitivity test underscores challenges such as overpricing and underpricing, particularly when options approach maturity. Furthermore, the study indicates that optimizing the neural network structure enhances overall model performance. These findings provide valuable insights for practitioners in the financial domain leveraging machine learning models for option pricing.

Executive Summary

In the dynamic and volatile equity derivative market, the pricing of American call options requires models that are both fast and accurate due to swift and unpredictable fluctuations in underlying security prices. Traditionally, numerical simulation methods like binomial trees and Monte Carlo simulations have been employed for American options, but the advent of machine learning has introduced alternative techniques.

This study investigates the application of a Gradient Boosting Neural Network for pricing American call options, utilizing a Heston stochastic volatility model with a Partial Differential Equation (PDE)-based pricing scheme to generate realistic data. The primary objectives are to explore the impact of dataset size on model performance, compare four models (Decision Tree, Gradient Boosting Tree, Neural Network, and Boosting Neural Network), conduct a sensitivity test on the Boosting Neural Network, and attempt model tuning for performance enhancement.

Key Findings:

1. Optimal Dataset Size: Increasing the dataset size improves model performance, reducing prediction errors. However, benefits plateau beyond a dataset size of 100 thousand, suggesting an optimal point for practitioners.
2. Model Performance: Neural Networks outperform Tree-based models in terms of both accuracy and speed. Boosting proves beneficial in reducing errors across all models, highlighting its effectiveness as an ensemble technique.
3. Sensitivity Test: The Boosting Neural Network exhibits challenges such as overpricing and underpricing, particularly as options approach maturity. This underscores the importance of understanding model limitations and potential biases.
4. Model Tuning: Optimizing the neural network structure enhances overall model performance, emphasizing the significance of fine-tuning in leveraging machine learning for option pricing.

Implications for Practitioners:

These findings provide valuable insights for practitioners in the financial domain leveraging machine learning models for option pricing. Understanding the optimal dataset size, the superiority of Neural Networks over Tree-based models, and the effectiveness of boosting techniques can guide decision-making in adopting and enhancing pricing models. Additionally, awareness of the challenges and limitations of the Boosting Neural Network underscores the need for careful consideration and continuous refinement in real-world applications.

In conclusion, this study contributes to the ongoing discourse on the application of machine learning in finance, offering practical insights for those navigating the complexities of option pricing in a rapidly changing market environment.

Introduction

The equity derivative market is known for its fluctuation. Swift and unpredictable fluctuations in the underlying security prices, coupled with various pricing parameters, can transpire suddenly, often driven by company news and the dynamic interplay of macroeconomic, financial, and political factors. This environment calls for a pricing model that is both fast and accurate.

Traditionally, American options, due to their uncertain exercise time, are priced by numerical simulation methods like binomial tree and Monte Carlo simulation. Since the dawn of machine learning, people have started to use machine learning techniques on option pricing. Methods like support vector machine (SVM) and tree-based models are proven to be efficient doing option pricing.B.V. Phani; B. Chandra; Vijay Raghav(2011) used SVM to

price different stock options and came up with positive results. Nowadays, deep learning and neural network are getting more and more popular. Researchers are exploring the possibility of using neural network in option pricing. David Anderson1 and Urban Ulrych(2023) compared the performance of the neural network with PDE numerical method and Monte Carlo Method pricing American call option. The study found that neural network outperform other models in both speed and precision. Also, it mentions that ensemble techniques may enhance the performance of neural network.

In our study, the first thing we want to explore is whether neural networks can beat traditional machine learning methods. We choose tree-based models for they are well known and easy to apply. Then we want to improve the neural network by using ensemble techniques, to be specific, boosting. Boosting is a machine learning ensemble technique that combines the predictions from multiple weak learners to create a strong learner. The basic idea behind boosting is to sequentially train a series of weak models, each focusing on the mistakes made by its predecessors, and then combine their predictions in a weighted manner. This helps improve the overall predictive performance of the ensemble. To make a comparison study, we trained 4 models, decision tree, gradient boosting tree, neural network and boosting neural network. We do comparison in 2 dimensions, tree-based model against neural network and plain model against boosting model.

Data used in this study is generated by Heston stochastic volatility model with a PDE-based pricing scheme. The advantage of this method is that it has non-constant volatility so the result simulates reality better. We randomly pick market parameters and volatility parameters from certain ranges and compute American call option prices accordingly. Then we apply the Feller condition and non-negativity. After getting the dataset, we experiment on the influence dataset size has on model performance. It gives an idea about the optimal dataset size to experiment on.

After setting up the dataset, we fit the four models, decision tree, gradient boosting tree, neural network and boosting neural network. We compare the performance of these models. In our study, we use mean absolute error (MAE) as error criteria. We compare both MAE and prediction time. Then we do a sensitivity test on boosting neural network to see its strength and weakness, and hyper tuning to further improve the model.

Exploratory Data Analysis

In our study, 2,110,000 samples are generated using the Heston stochastic volatility model with a PDE-based pricing scheme. Unlike the Black-Scholes-Merton model that assumes volatility to be constant over the entire life of an option, the Heston stochastic volatility model allows us to account for the dynamic of implied volatility, which is more realistic in pricing an American-style option.

The definition of the Heston stochastic volatility model is to consider asset price (S_t) and variance (v_t) at time t under the following stochastic differential equations (SDEs).

$$dS_t = (rf - q)S_t dt + \sqrt{v_t} S_t dW_t,$$
$$dv_t = \boldsymbol{\kappa}(\theta - v_t)S_t dt + \sigma\sqrt{v_t} dZ_t,$$
$$dZ_t dW_t = \rho dt$$

Where $\boldsymbol{\kappa} > 0$ is mean reversion rate of the variance

$\theta > 0$ is long-run average variance

$-1 \geq \rho \geq 1$ is correlation between the underlying asset prices and volatility processes

$S_t > 0$ is asset price at time t

$v_t > 0$ is variance at time t

The parameters to generate each sample are the initial price of the underlying asset (S0), strike price (K), time to expiry (T), risk-free rate (rf), dividend yield (q), initial variance (v0), long-run average variance (θ), correlation between the underlying asset prices and volatility processes (ρ), mean reversion rate of the variance ($\boldsymbol{\kappa}$) and volatility of the volatility (σ). As 1 set of the parameters can generate 1 price sample, we randomly generate the sets of parameters using uniform random distributions except the initial price S0 which is fixed to 100, and the ranges of each parameters is show in **TABLE a**

TABLE a The range of the parameters

Parameter	Uniform distribution U(lower bound, upper bound)	
	Lower bound	Upper bound
strike price (K)	90	110
time to expiry (T)	1/255	1.
risk-free rate (rf)	0.01	0.06
dividend yield (q)	0.01	0.03
initial variance (v0)	0.01	0.41
long-run average variance (θ)	0.01	0.41
correlation between the underlying asset prices and volatility (ρ)	-0.9	0.9
mean reversion rate of the variance ($\boldsymbol{\kappa}$)	0.0	5.0
volatility of the volatility (σ)	0.01	1.0

After the sample generation, we derive moneyness, which is a crucial and popular feature in analyzing and computing option prices, using the following formula.

$$Moneyness = \frac{Strike\ price\ (K)}{Initial\ price\ (S0)}$$

The Feller condition ($2\boldsymbol{\kappa}\theta > \sigma^2$) is then applied to guarantee positive volatility, and outliers are removed to improve the data quality. The number of remaining samples is 1,582,935. **Figure a** shows the distributions of each parameter after applying the described conditions.

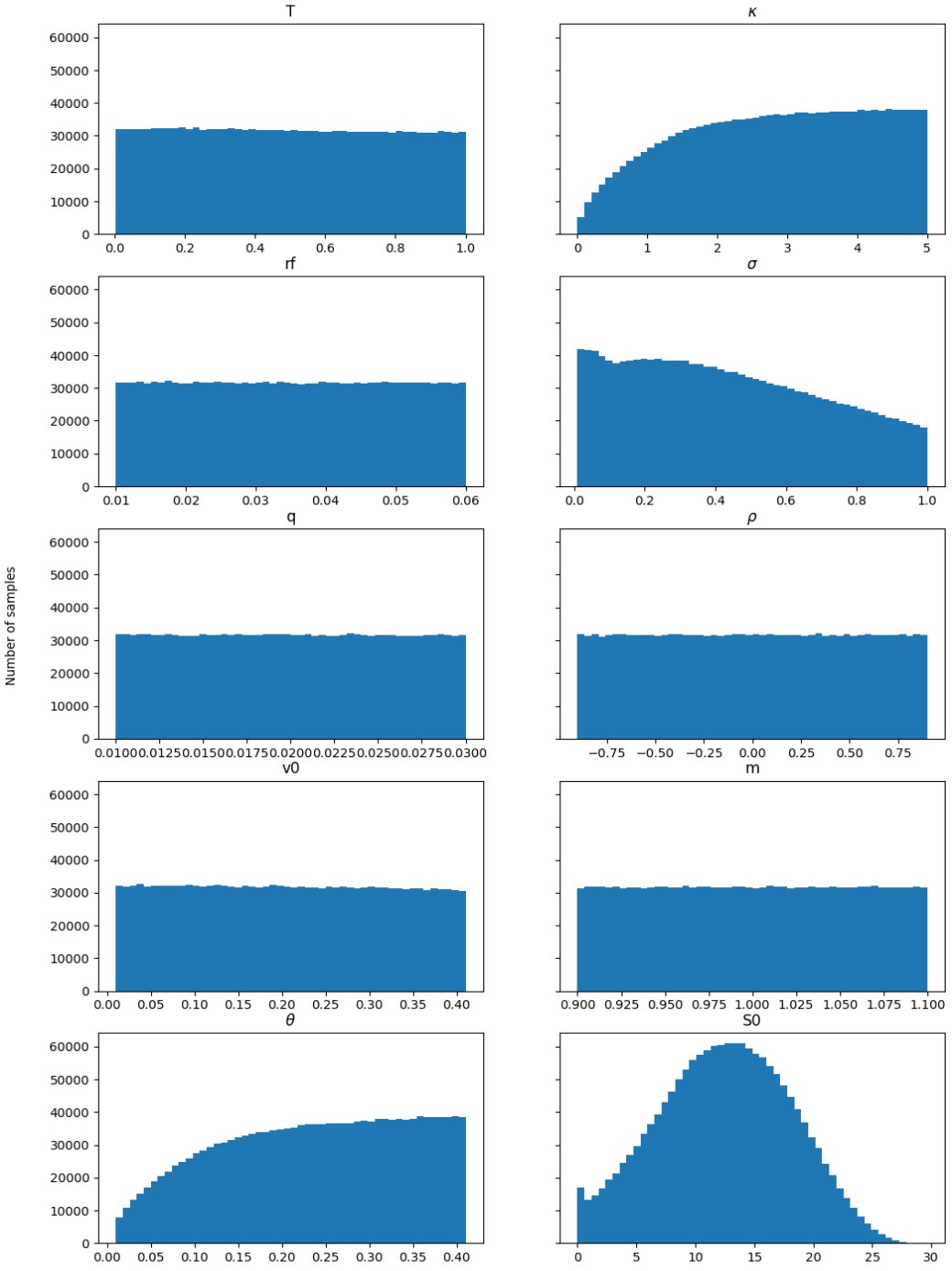

Figure a The distributions of each parameter

Figure b shows the relationship between the moneyness (m) and initial price (S0) after applying the described conditions.

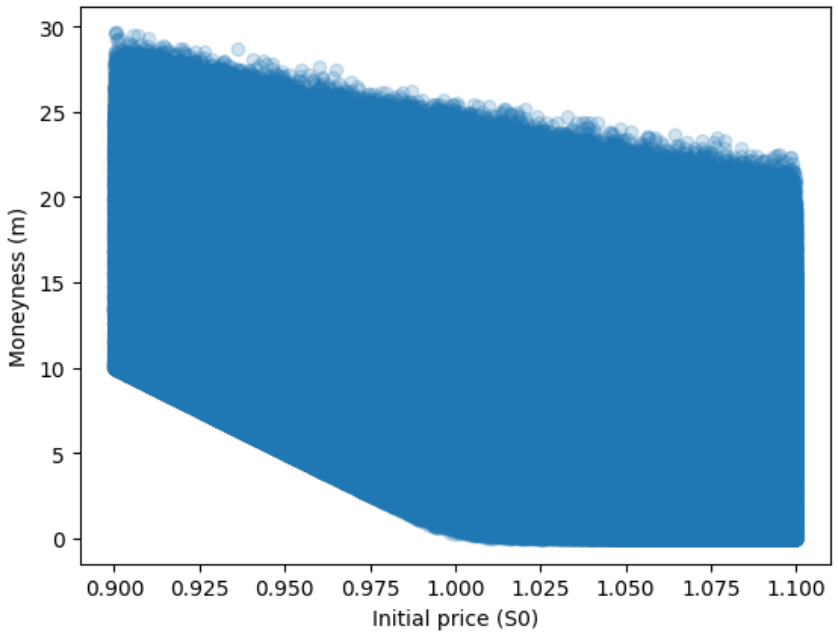

Figure b The relationship between moneyness (m) and initial price (S0)

Methodology

1.Decision tree model

Tree-based models are a class of machine learning algorithms that use decision trees as their fundamental building blocks for making predictions. Decision trees are hierarchical structures where each node represents a decision based on a particular feature, leading to a split in the data. The final predictions are made by the leaves of the tree.

A decision tree is a fundamental machine learning model that takes the form of a hierarchical tree structure, where each node represents a decision based on a specific feature, and each branch represents a possible outcome of that decision. Figure c is an example of decision tree structure. The tree structure is recursively built during the training process, with nodes determined by features that best separate the data.

2.Gradient Boosting Trees (GBT)

Gradient Boosting Trees is a powerful ensemble machine learning technique that sequentially builds a series of decision trees, each correcting the errors of the preceding ones. GBT minimizes a specified loss function by fitting each tree to the negative gradient of the loss with respect to the predictions. During training, the model places greater emphasis on instances with larger residuals, enabling the ensemble to focus on difficult-to-predict data points. This iterative process results in a highly adaptive and accurate predictive model. GBT is known for its ability to capture complex relationships, handle non-linear patterns, and provide feature importance insights.

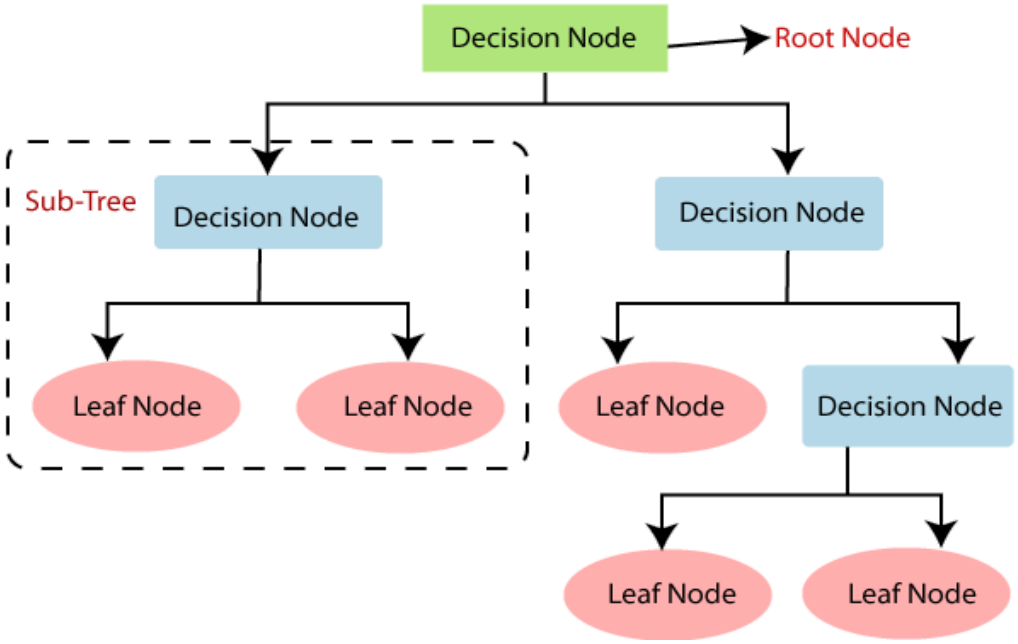

Figure c Structure of a Decision Tree

3.Deep Neural Networks (DNNs)

Deep Neural Networks are one of the most powerful machine learning models recently. It usually consists of 3 types of neural network layers, which are the input layer, hidden layers and output layers. **Figure d** shows an example of a deep neural network with 1 input layer, 4 hidden layers and 1 output layers. Each layer has a certain number of neurons, which apply a linear function to all the features obtained from the preceding layer and then apply an activation function to obtain a final output feature, which is used as a feature of the next layer. **Figure e** illustrates the described computation procedure inside 1 neuron.

Although there are many types of neural network layers, such as convolutional and recurrent layers, we use the fully-connected layers for all layers in our study as they best suit our application.

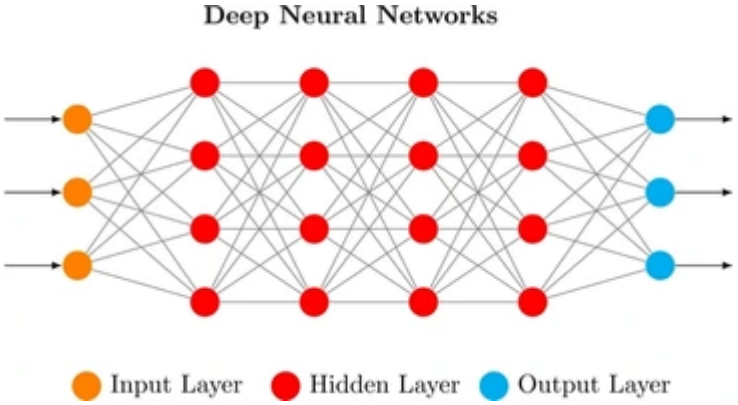

Figure d Structure of deep neural networks

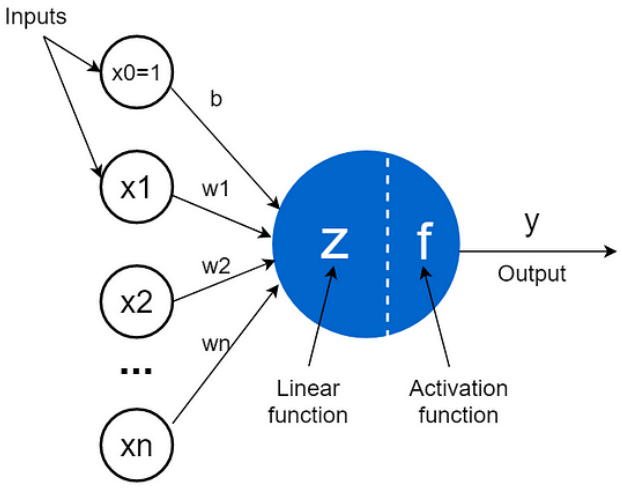

Figure e Computation inside 1 neuron

4. Gradient Boosting Neural Network (GBNN)

Like GBT described before, Gradient Boosting Neural Networks is an ensemble technique, which trains a neural network using a set of features and targets, and then uses the error collected from the preceding models' prediction as a target of the next neural network model. **Figure f** illustrates how a gradient boosting neural network is trained.

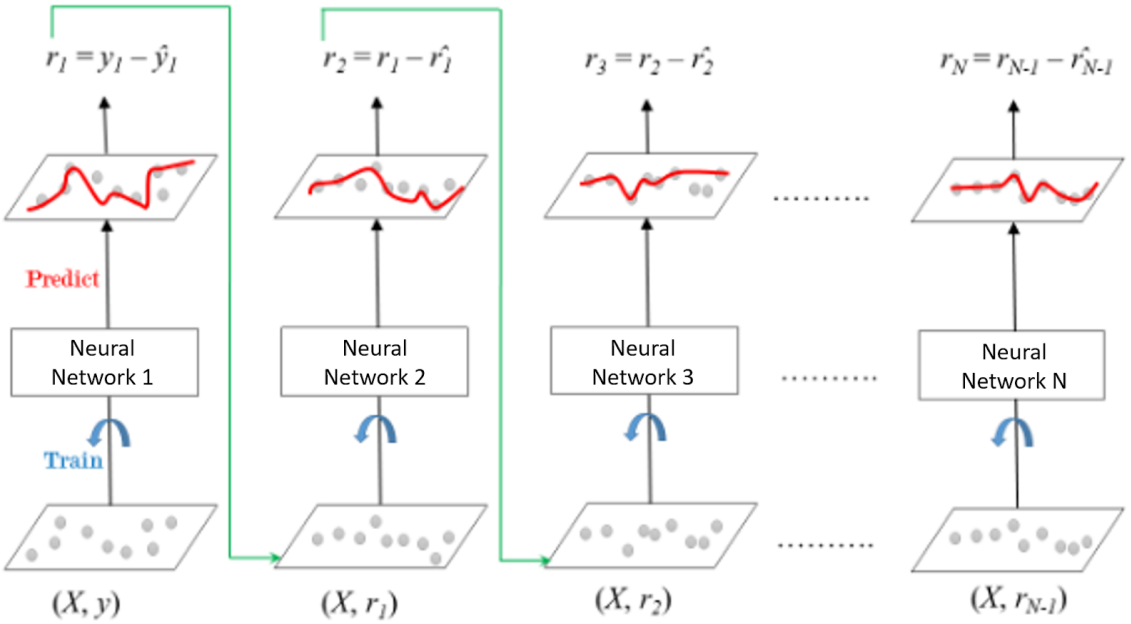

Figure f How a gradient boosting neural network is trained

Model fitting and evaluation

In this study, we conducted 3 experiments. First, we experimented the influence of dataset size on model performance. After having the optimal dataset, we fit 4 models on it, then test

and compare the MAE and prediction time. Lastly, we conducted a sensitivity test to find out the strengths and weaknesses of the boosting neural network.

1.Model fitting and prediction

Algorithm a shows how we use each model to price American call options. For each model, the inputs are r, q, v0, σ, κ, θ, ρ, M, and T. The output is the current price for an American call option. We first evaluate American option prices using a chosen standard pricing method across an entire parameter space Ψ. Then we train the machine learning model on the dataset. Then we use the trained model to price American call options.

Algorithm a Machine Learning-based American option pricer under stochastic volatility

Input: Chosen parameter ranges for r, q, v0, σ, κ, θ, ρ, M, and T..
Output: Current price of an American call option per underlying applicable in a market-making environment.

(a) Training data generation: Evaluation of American option prices using a chosen standard pricing method across an entire parameter space Ψ.
(b) Learning: Offline training of a machine learning model over a parameter-price data set.
(c) Pricing: Application of the machine learning pricer in a market-making environment

For tree models, we did hyperparameter tuning. For the decision tree, we tuned the maximum depth in the range of 5 to 50, minimum samples per leaf in [1,2,4,8] and minimum sample split in [2,5,10,20]. For the gradient boosting tree, we tuned maximum depth form 3 to 15, and number of estimators from 100 to 300. The optimal parameters for the decision tree is 35 as maximum depth, at least 4 samples per leaf and at least 5 samples to split. For gradient boosting trees, the parameters are 7 as maximum depth and 300 estimators. For neural network model structure, we use 1 input layer followed by 3 hidden layers and 1 output layer where each layer has 40 neurons with Leaky ReLU activation function with leaky slope of 0.01. Adam optimizer with the initial learning rate of 0.005 and the batch size of 4096 samples is used during all the training. Early stopping with the patient of 50 epochs is also used to terminate the training if the test loss stops reducing. The learning rate schedule with the patient of 20 epochs is also used to reduce the learning rate by 10 times when the training loss stops reducing.

2.Dataset Size

To determine the optimal dataset size, we trained the decision tree, gradient boosting tree and neural network on different data sizes, specifically 10000, 20000, 40000, 80000, 100000, 200000, 400000, 800000, 1000000 and 1200000. For each size, we trained a model

and got a result of MAE. Figure d shows the result of the size test.

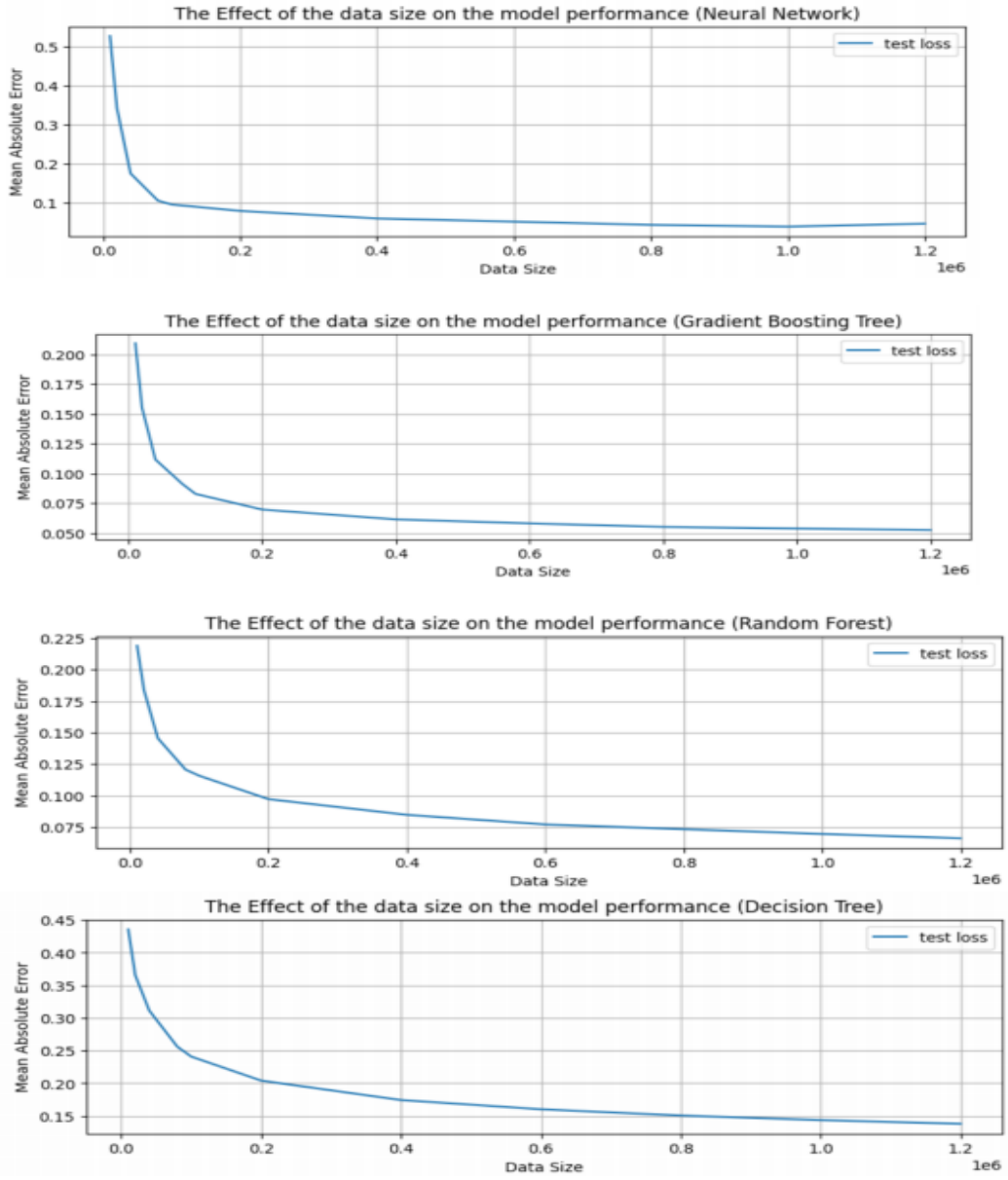

Figure g The relationship between dataset size and model prediction MAE

As the figure shows, when dataset size increases, the MAE of trained models decreases. However, after dataset size exceeds 100 thousand, the reduction in MAE becomes insignificant. Considering the time consumption of generating data and model training, we believed that using a train dataset at 1.1 million, testing and validation set at 230 thousand is optimal for further experiment.

3. Comparison experiment

After having the optimal dataset, we fitted the models on the training set and tested on the test set for prediction MAE and time.

	MAE	Test time [second] (237,440 samples)	Prediction time [micro second/sample]
Decision Tree	0.449475	0.078001	0.33
Gradient Boosting Tree	0.148981	1.74299	7.34
Plain Neural Network	0.0263	0.01605**	0.068
Gradient Boosting Neural Network (with 1 correcting model)	0.0226	0.0469**	0.20
Gradient Boosting Neural Network (with 2 correcting models)	0.0215	0.109**	0.46
Gradient Boosting Neural Network (with 3 correcting models)	0.0214	0.135**	0.57

Table b Results of 4 models on test set.

Table b presents the result of 4 models. As shown in the table, neural networks outperform tree-based models in both accuracy and prediction speed. Gradient boosting method does improve the accuracy of both tree model and neural network. However, only Gradient Boosting Neural Networks with 2 correcting models exhibit a significant reduction in the MAE. Gradient Boosting Neural Networks with 2 correcting models is overall the best model, considering the accuracy and prediction time.

4. Sensitivity test

The purpose of this experiment is to show how well the gradient boosting neural network with 1 correcting model performs when only 1 parameter varies while others are fixed when compared to the target price obtained from the Heston stochastic volatility model with the PDE-based pricing scheme.

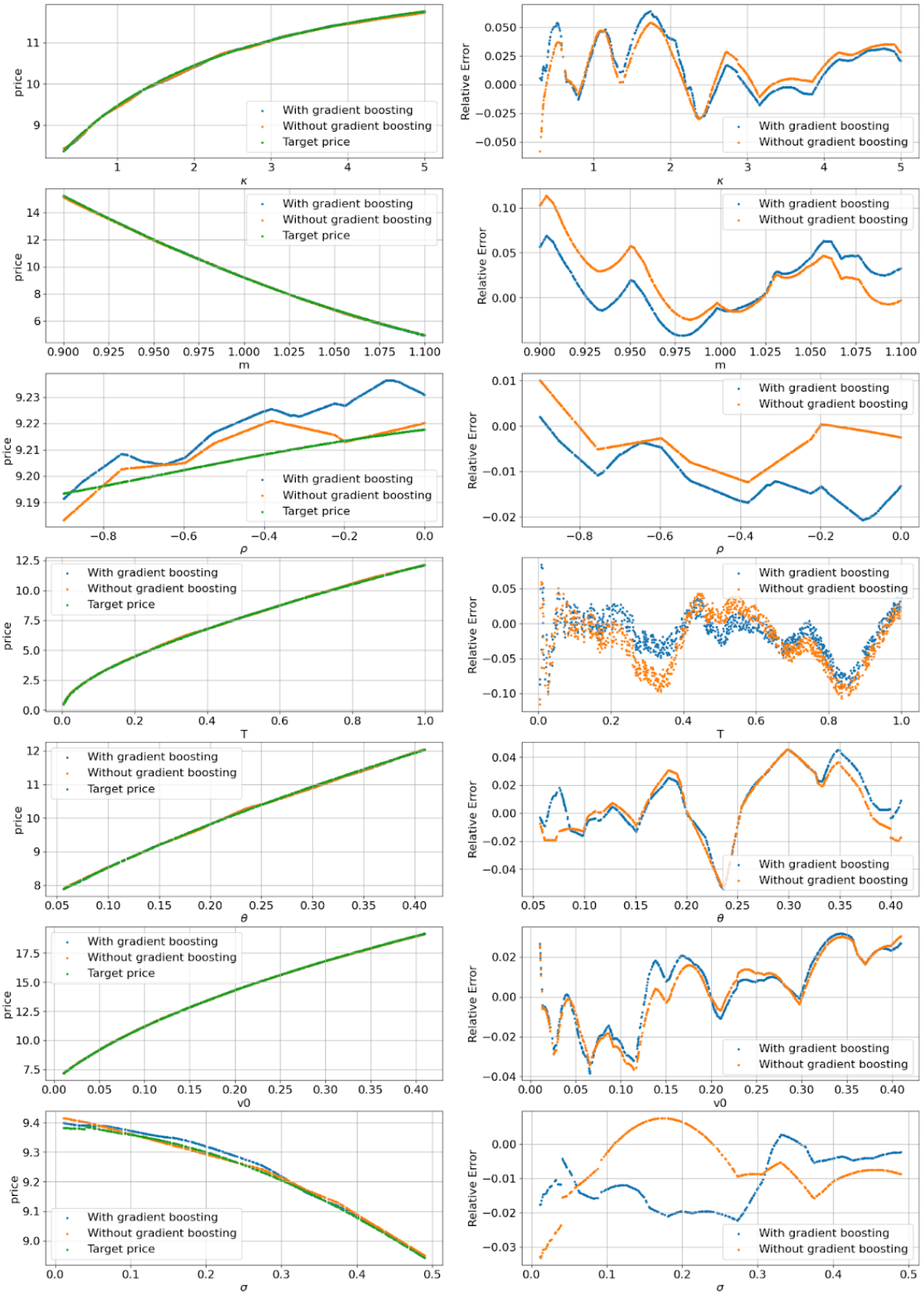

Figure h Sensitivity test result on Gradient Boosting Neural Network

From **Figure h**, we see the comparison between neural network models with and without gradient boosting technique. Most cases with the extreme relative errors, whose magnitudes are greater than 5 cents, from the model without gradient boosting are mitigated by the model with gradient boosting as it can be clearly seen in the cases of kappa (κ), moneyness (m) and time to maturity (T) in **Figure h**.

Hyperparameter tuning on neural network model structure

We investigate if the proposed neural network structure is optimal or not. Due to limited results, we reduce the size of the training set to 100,000 samples with a maximum number of epochs of 1,500. The number of hidden layers and the number of neurons of each layer are varied, and the final MAEs are then compared as shown in **Table c**.

Table c The comparison of model performance with different model structures

Number of layers	Number of neurons	Training MAE	Test MAE
4	20	0.167	0.168
3	20	0.155	0.156
2	20	0.154	0.155
4	40	0.082	0.083
3	40	0.069	0.070
2	40	0.055	0.056
4	60	0.064	0.065
3	60	**0.047**	**0.049**
2	60	0.050	0.052

According to **Table c**, the highlighted model structure with number of layers of 3 and number of neurons of 40 is our proposed neural network model structure. The model structure with number of layers of 3 and number of neurons of 60 yields better MAE in both training and test datasets. The results therefore leave some room for future improvement to explore better model structure.

Results and Conclusions

This study focused on using the gradient boosting neural network to price American call options. Heston stochastic volatility model with a PDE-based pricing scheme was used to generate data. We first tested the influence of dataset size on model performance to get the optimal dataset size. Then we fitted 4 models, decision tree, gradient boosting tree, neural network and boosting neural network and compared their prediction MAE and time.

Furthermore,we did a sensitivity test on the gradient boosting neural network to see its strength and weakness and tuned the model, trying to improve it.

Our experiment results demonstrated the following findings. First, increasing dataset size can improve the model performance by reducing the prediction error. However, benefits from large data size begin to vanish when data size exceeds 100 thousand. Secondly, neural networks outperforms Tree-based models in terms of accuracy and speed. On top of that, boosting does help the models by reducing errors. However, the sensitivity test points out that even for the gradient boosting neural network, overpricing and underpricing are hard to ignore in certain regions, for example, when the option is close to maturity. Also, it is shown that better structure of the neural network can improve the performance.

References

Anderson, David and Ulrych, Urban, Accelerated American Option Pricing with Deep Neural Networks (December 30, 2022). Quantitative Finance and Economics, 2023, 7(2): 207-228. doi: 10.3934/QFE.2023011.

B. V. Phani, B. Chandra and V. Raghav, "Quest for efficient option pricing prediction model using machine learning techniques," *The 2011 International Joint Conference on Neural Networks*, San Jose, CA, USA, 2011, pp. 654-657, doi: 10.1109/IJCNN.2011.6033283.