Comparative Analysis of Deep Learning and Ensemble Models for Pricing American Call Options Across Varying Sample Sizes

Ben Granados
Dhruv Oza
Goldi Pangayom
Kegen Qian

Abstract

1. Evaluating advanced machine learning models in option pricing contributes to the development of more sophisticated and reliable financial tools.

2. This progress is essential for investors and financial institutions to make more informed decisions, reducing risks associated with mispriced assets and contributing to a more transparent and efficient market environment.

3. This study presents a comprehensive comparison between deep learning models, specifically neural networks, and ensemble models, including Gradient Boosting and XGBoost, in the context of pricing American Call Options.

4. Leveraging the Heston model parameters for data generation, the research focuses on evaluating the performance of these models under varying sample sizes.

5. Key aspects of comparison include computational efficiency and pricing accuracy.

6. The neural networks are explored for their capability in capturing complex patterns, while the ensemble models are examined for their robustness in diverse dataset handling.

7. The findings aim to shed light on the suitability and effectiveness of advanced machine learning techniques in the dynamic and complex domain of financial option pricing.

Research Outline

Data Generation	Model Training and Prediction	Model Comparison	Conclusion
• Generated American call option prices using Heston Parameters. • Cleaned the Data • Generated datasets of 3 different sizes.	• Trained the different models for the 3 different datasets. • Hyperparameter Tuning	• Compared the prediction time and root mean squared error across different sample sizes	• Found the model that trades off the accuracy with the prediction time in the best possible way

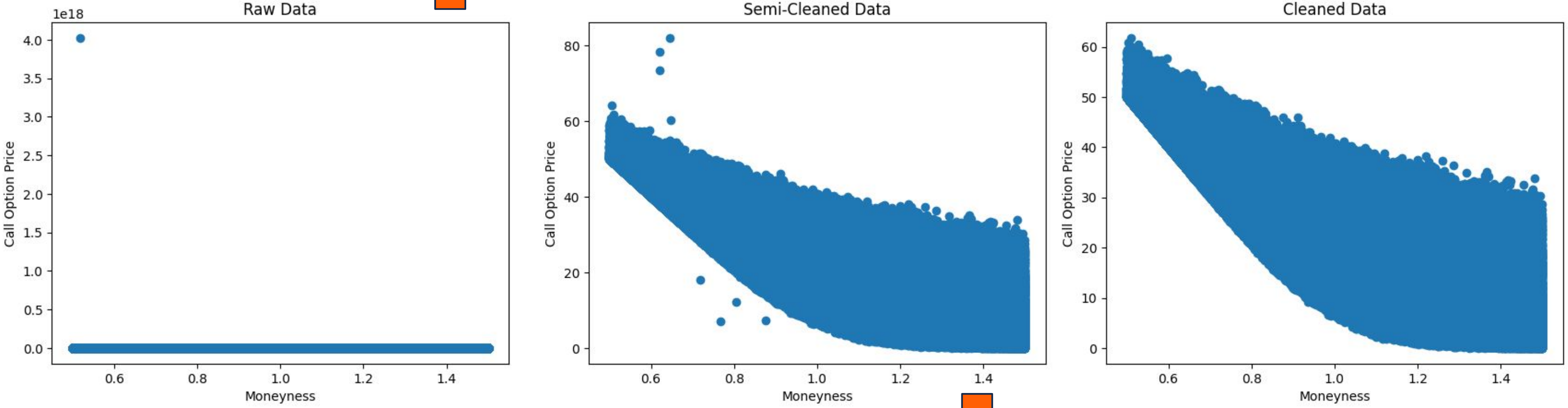
Data Generation Process

Generating American Call Prices → Data Cleaning → Dataset Generation

- **Methodology:** Generated three separate datasets using the Heston model for American Call Option pricing. This involved creating datasets with different sizes, each using uniformly randomly distributed Heston parameters.
- **Adherence to Feller Condition:** Ensured that the selected Heston model parameters satisfied the Feller condition across all datasets. This was crucial to maintain the stability of the model and ensure the accuracy of the generated option values. The Feller condition: $2\kappa\theta$ >= σ^2.
- **Dataset Sizes**:
  - The first dataset comprised 2,000 data points.
  - The second dataset was larger, consisting of 50,000 data points.
  - The third and largest dataset contained 250,000 data points.
- **Purpose of Varied Sizes:** The different sizes of datasets aimed to test the models' performance and scalability. This approach allowed for a thorough assessment of how well each model adapts to datasets of varying magnitudes, which is critical for real-world applications where data size can significantly vary.

Data Cleaning and Dataset Generation

Removing extreme values



Removing visible outliers

- Used the cleaned data to create three distinct datasets with sizes of 2000, 50000, and 250000 instances, which were then employed to train the models

Our Models

Overview of Models:

1. **Feed-Forward Neural Network:**

- Structure: This model consists of multiple layers of neurons, each layer fully connected to the next. It is designed to recognize complex patterns and relationships in data.
- Functionality: By processing inputs through its layers, the network learns to map the relationship between the market conditions described by the Heston model parameters and the option prices.
- Advantage: Neural networks are known for their ability to learn nonlinear relationships, making them particularly suitable for modeling the intricacies of financial markets.

2. **Gradient Boosting:**

- Structure: Gradient Boosting is an ensemble learning technique that builds models sequentially, each new model correcting the errors made by the previous ones.
- Functionality: In this context, it uses decision trees as base learners, which are aggregated to improve the model's ability to predict option prices accurately.
- Advantage: The model excels in handling varied datasets and reducing bias, making it robust against overfitting, especially in scenarios with complex market dynamics.

3. **XGBoost (eXtreme Gradient Boosting):**

- Structure: XGBoost is an advanced implementation of gradient boosting with enhanced performance and speed. It is optimized for computational efficiency and model performance.
- Functionality: It includes built-in regularization which helps in reducing overfitting, and it is capable of handling sparse data, making it versatile and powerful.
- Advantage: Given its high computational efficiency and accuracy, XGBoost is particularly suited for scenarios where speed and precision in option pricing are critical.

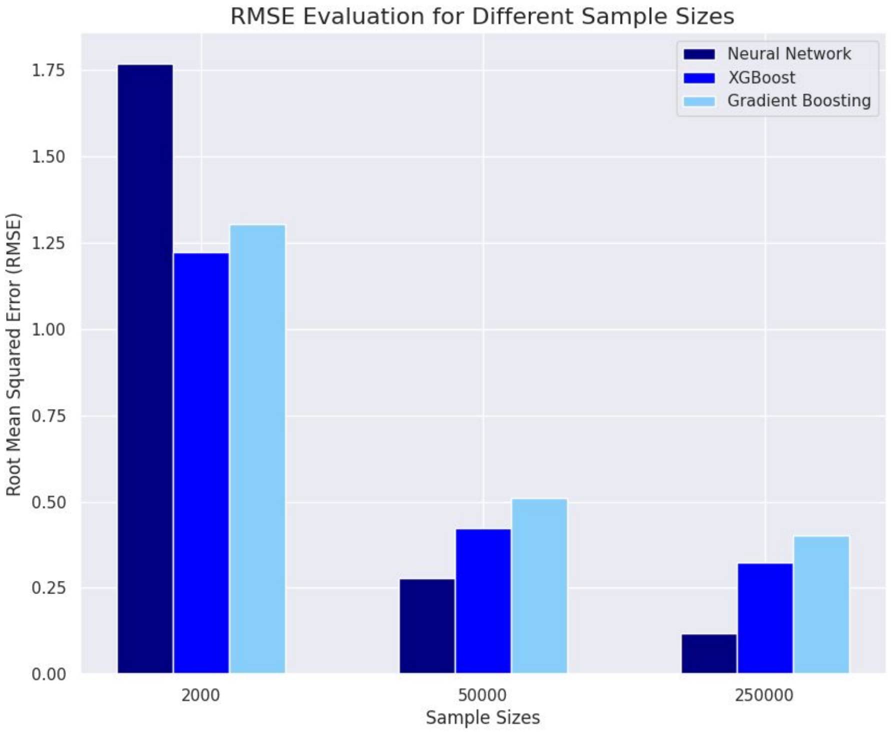
Hyperparameter Tuning

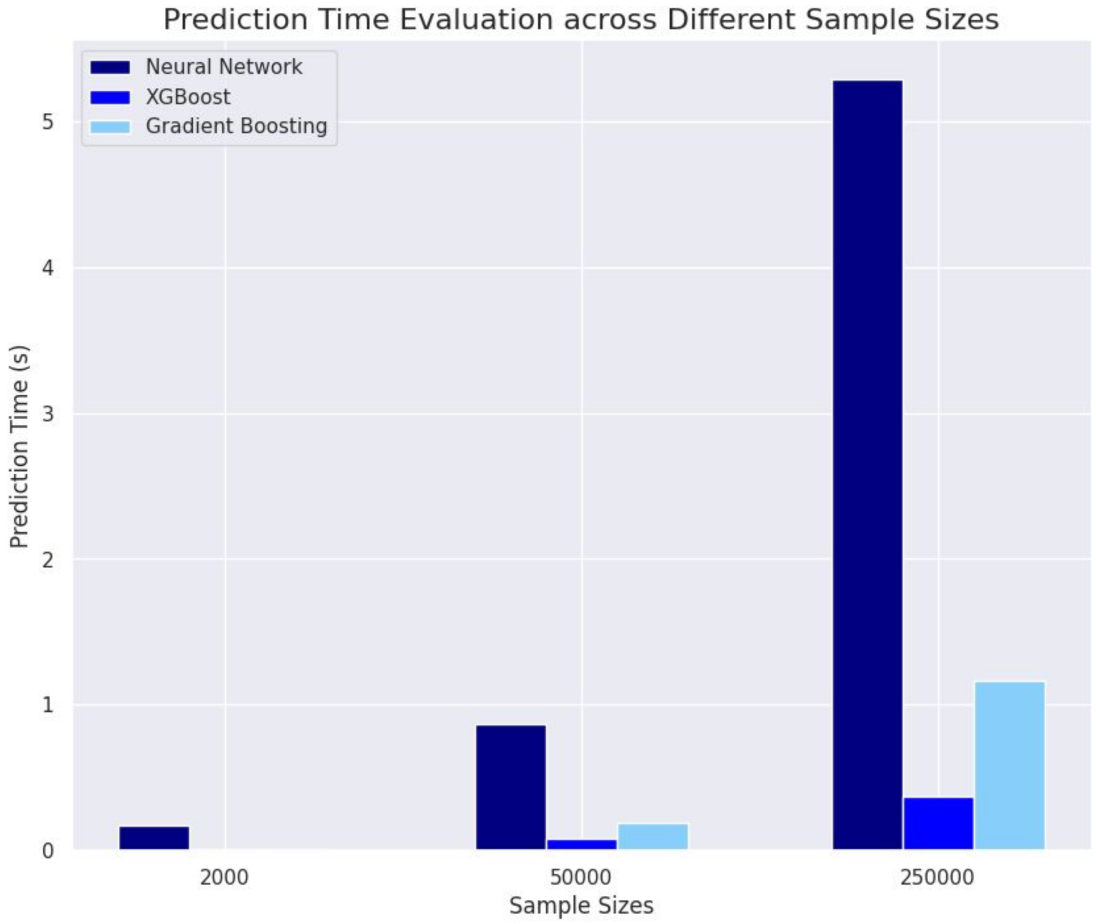
- **Gradient Boosting** - we used a GridSearch to find the optimal hyperparameters for our model. We chose to tune: max_depth, n_estimators, and max_features.

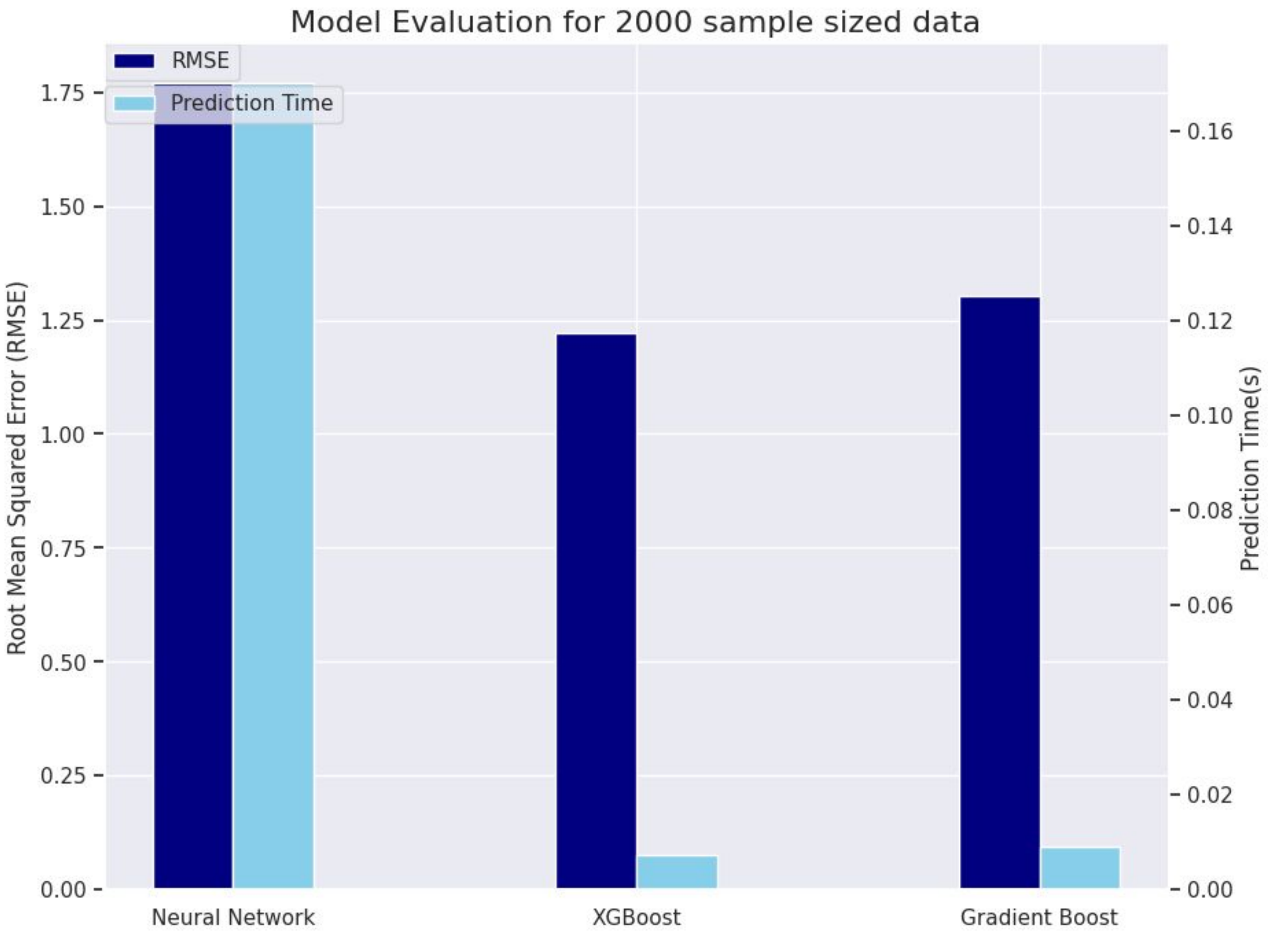
- **XGBoost** - we used GridSearch to find the optimal hyperparameters for our model. We chose to tune: max_depth, n_estimators, and learning_rate.

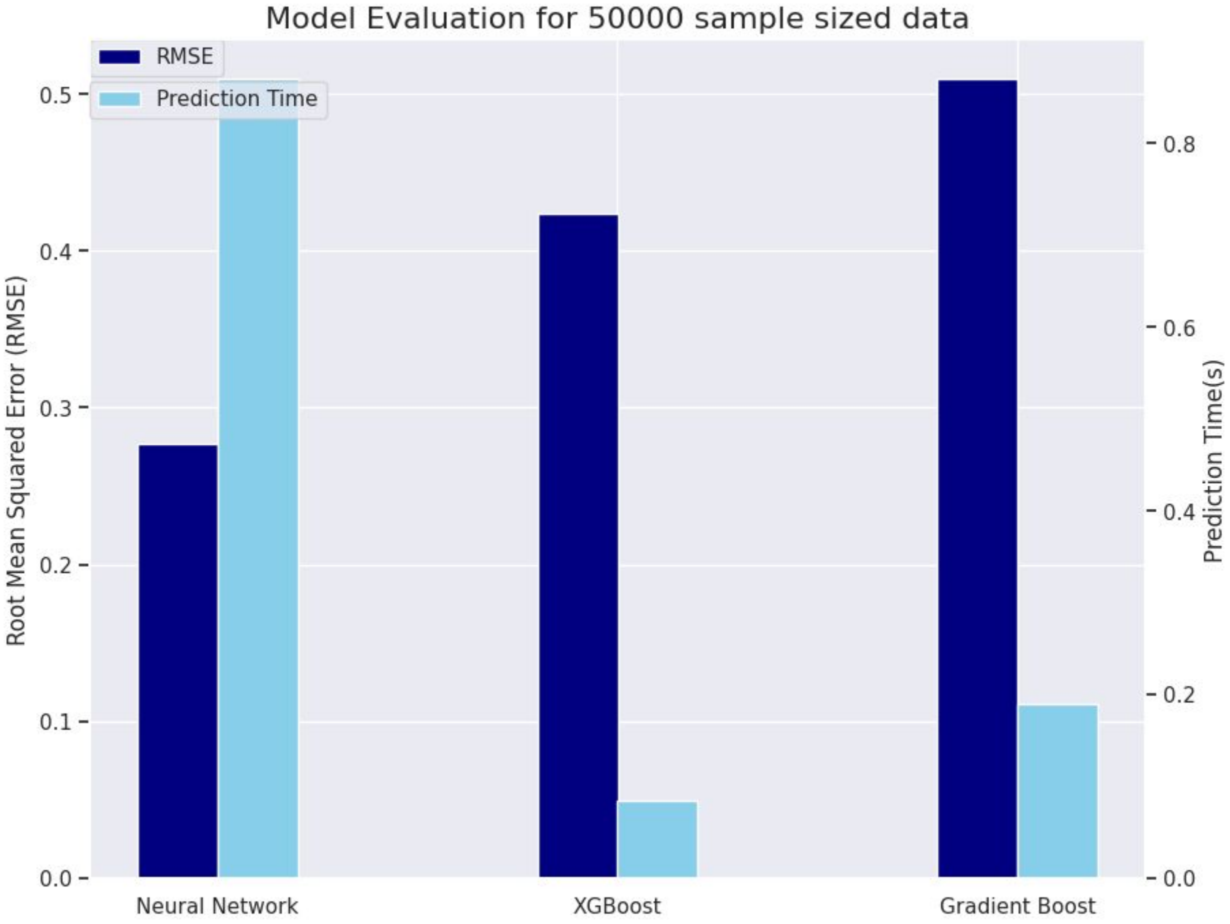
- **Neural Network** - we tuned the optimizers, learning rate, number of hidden layers, number of neurons to create model.

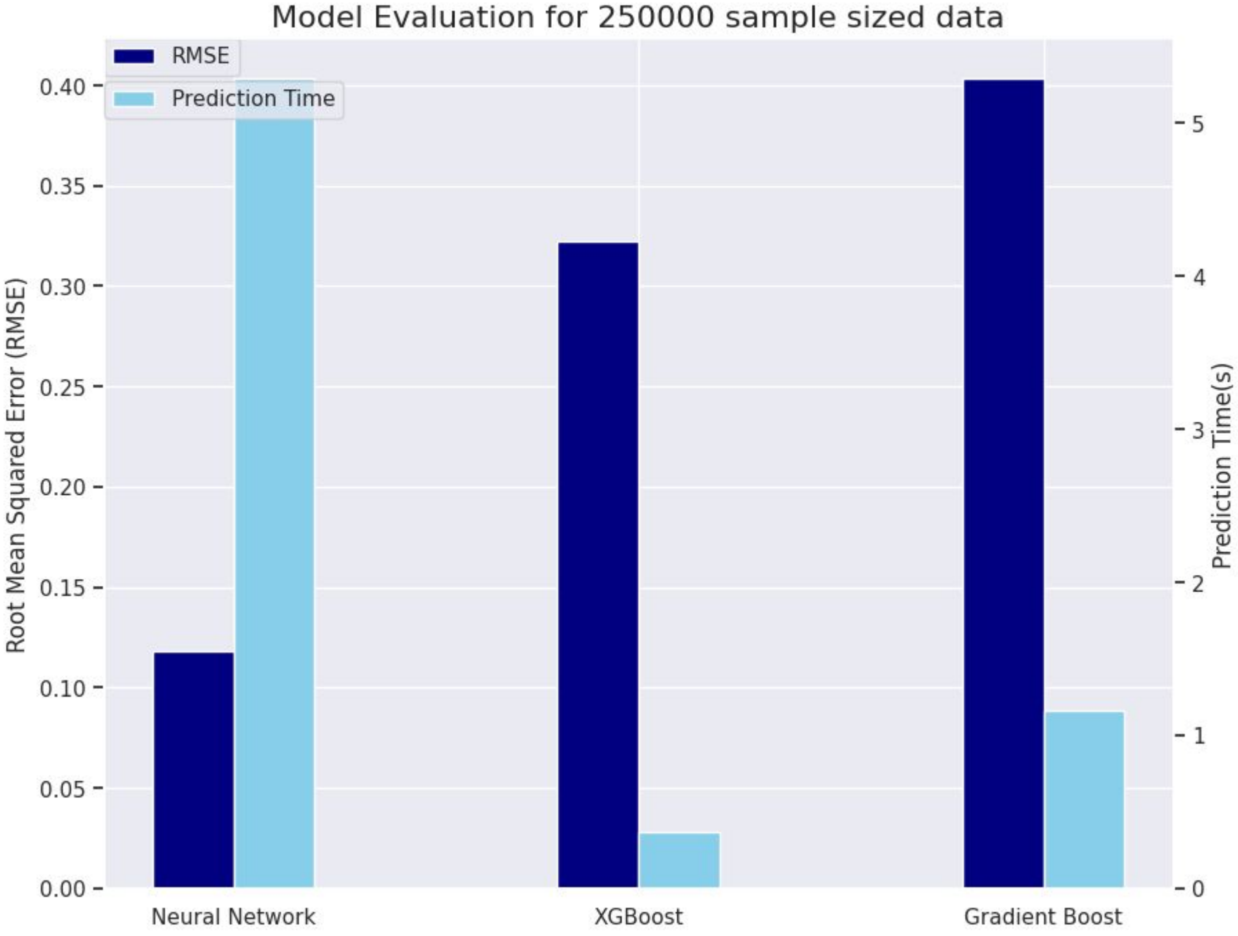
Model Comparison




Prediction Time Evaluation across Different Sample Sizes
Neural Network
XGBoost
Gradient Boosting
Prediction Time (s)
0
1
2
3
4
5
2000
50000
250000
Sample Sizes


Model Evaluation for 2000 sample sized data
RMSE
Prediction Time
Root Mean Squared Error (RMSE)
Prediction Time(s)
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
Neural Network
XGBoost
Gradient Boost


Model Evaluation for 50000 sample sized data
RMSE
Prediction Time
Root Mean Squared Error (RMSE)
Prediction Time(s)
0.0
0.1
0.2
0.3
0.4
0.5
0.0
0.2
0.4
0.6
0.8
Neural Network
XGBoost
Gradient Boost



Model Evaluation for 250000 sample sized data
RMSE
Prediction Time
Root Mean Squared Error (RMSE)
Prediction Time(s)
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0
1
2
3
4
5
Neural Network
XGBoost
Gradient Boost

Accuracy - Time Tradeoff

Question- How we calculated the trade-off metric value?

Trade-Off Metric = $\frac{1}{RMSE \times TestTime}$

Properties of this metric:

- It is inversely proportional to the testing error as well as the prediction time.
- Punishes high values of RMSE.
- Punishes high prediction time.
- More the value, the better the model.

Sample Size	Model	Accuracy vs Time Trade Off Value
2000	Neural Network	3.322852
	XGBoost	111.868864
	Gradient Boost	85.147426
50000	Neural Network	4.145904
	XGBoost	27.818509
	Gradient Boost	10.377044
250000	Neural Network	1.601424
	XGBoost	8.350455
	Gradient Boost	2.132176

Overview of Model Advantages

- **Overall Best for Accuracy-Time Trade-off**:
    - XGBoost:
        - Fastest in computation time across all sample sizes.
        - Highest accuracy for the smallest sample size.
        - Overall best performer considering both accuracy and computation time.

- **Second Best for Accuracy-Time Trade-off:**
    - Gradient Boosting:
        - Faster than Neural Networks but slower than XGBoost in computation time for all sample sizes.
        - Superior to Neural Networks in accuracy for the smallest sample size.
        - Offers a balanced performance, but not as optimal as XGBoost.

- **Third in Accuracy-Time Trade-off:**
    - Neural Networks:
        - Slowest in computation time for all sample sizes.
        - Most accurate for medium and large sample sizes.
        - Ideal for larger datasets where accuracy is prioritized over speed.

Conclusion and Future Research Considerations

- **Key Takeaways:**
  - The choice of model depends on the specific requirements of accuracy and computational efficiency.
  - XGBoost is preferable for scenarios where both time efficiency and accuracy are crucial, particularly in smaller datasets.
  - Neural Networks are recommended for larger datasets where accuracy is more critical than computational speed.

- **Recommendations for future research:**
  - Consider applying feature engineering to generate more inputs for the models.
  - Try expanding the dataset to include a larger sample size for comparison.
  - Explore further hyperparameter tuning, that could significantly enhance the models' performance.