Summary of Neural Network for Forecasting Implied Volatility of Stock Options

Kamakshi Sarvananthan

Abstract. This paper explores the performance of a Multi-Layer Perceptron (MLP) neural network in forecasting the implied volatility of stock options. Implied volatility is a key metric in options trading, reflecting market expectations of future volatility. Therefore, accurate forecasting of implied volatility can significantly enhance trading strategies, enabling traders to make more informed decisions regarding option pricing, risk management, and the timing of trades. By anticipating changes in market volatility, traders can better adjust their positions to maximize profits and minimize risks, ultimately improving their overall trading performance.

1 Introduction

Implied volatility is an important metric in financial markets, particularly in options trading. It represents the market's expectations of future price volatility and plays a significant role in the pricing of options contracts. Accurate prediction of implied volatility can provide traders with a competitive edge, enabling more informed decision-making and optimized trading strategies. Traditional models for forecasting implied volatility include the Black-Scholes model, GARCH (Generalized Autoregressive Conditional Heteroskedasticity) models, and stochastic volatility models. However, traditional models such as the Black-Scholes model often face limitations due to their reliance on unrealistic assumptions.

This project leverages a Multi-Layer Perceptron (MLP) neural network for regression tasks to forecast the implied volatility of stock options, offering several advantages over traditional models. Neural networks can capture complex, non-linear relationships between input features and the target variable. The ability to predict implied volatility is particularly valuable for various options trading strategies. For instance, the short iron condor strategy, involves shorting a put and call and purchasing further out-of-the-money (OTM) puts and calls against that position. In this strategy, traders' profit if implied volatility falls when the stock price is between the short strikes. By accurately forecasting implied volatility, traders can better anticipate price movements and optimize these strategies for greater profitability.

The evaluation in this paper demonstrates the initial experimentation with the model's capability to predict implied volatility and lays the groundwork for further enhancements. Future improvements may include expanding the dataset, incorporating, or removing additional features, and exploring different neural network architectures to enhance prediction accuracy.

2 Data

The dataset consists of historical options data for tickers from 44 underlying companies, diversified by sector, market capitalization, and volatility, sourced from Yahoo Finance through the yfinance library. The features include Strike Price, Last Price, Open Interest, Moneyness, Contract Type, as well as Relative Strength Index, and the Middle Bollinger Band, both calculated using the TA-lib library. All null values were removed, resulting in approximately 3000 rows. The dataset was split into training, validation, and testing sets, with 80% allocated for training and validation, and 20% for testing. Categorical values were encoded numerically, and features were normalized to the range [0, 1] using min-max scaling, where each feature was transformed according to the formula:

$$normalized\ value = \frac{value - \min value\ of\ column}{range\ of\ column}$$

3 Model

3.1 Model Architecture

The model architecture utilizes a Multi-Layer Perceptron (MLP) neural network built with TensorFlow and Keras, designed specifically for regression tasks. The network begins with an input layer of 8 neurons, representing the input features. The input layer is followed by two hidden layers, each consisting of 64 neurons with Rectified Linear Unit (ReLU) activation functions. The final output layer of the model contains a single neuron that predicts the implied volatility. ReLU introduces non-linearity into the model. The activation function is defined as:

$$f(x) = max(0, x)$$

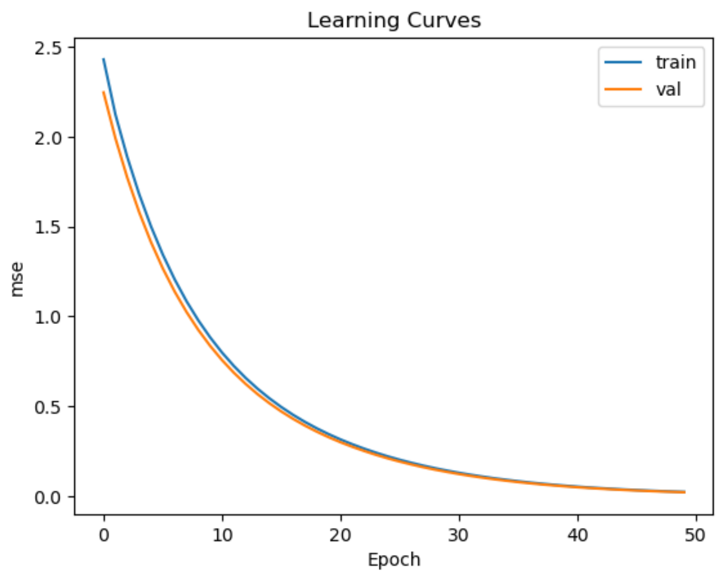
3.2 Compilation & Training

The model was compiled using the Adam optimizer with Mean Squared Error (MSE) chosen as the loss function and L2 regularization was applied to mitigate overfitting. The model was trained over 50 epochs with a batch size of 64. This setup ensures that the model has enough iterations to learn the underlying patterns in the data while the batch size strikes a balance between computational efficiency and convergence stability. The choice of 50 epochs is a starting point that can be adjusted based on the model's performance during training.

4 Performance Evaluation

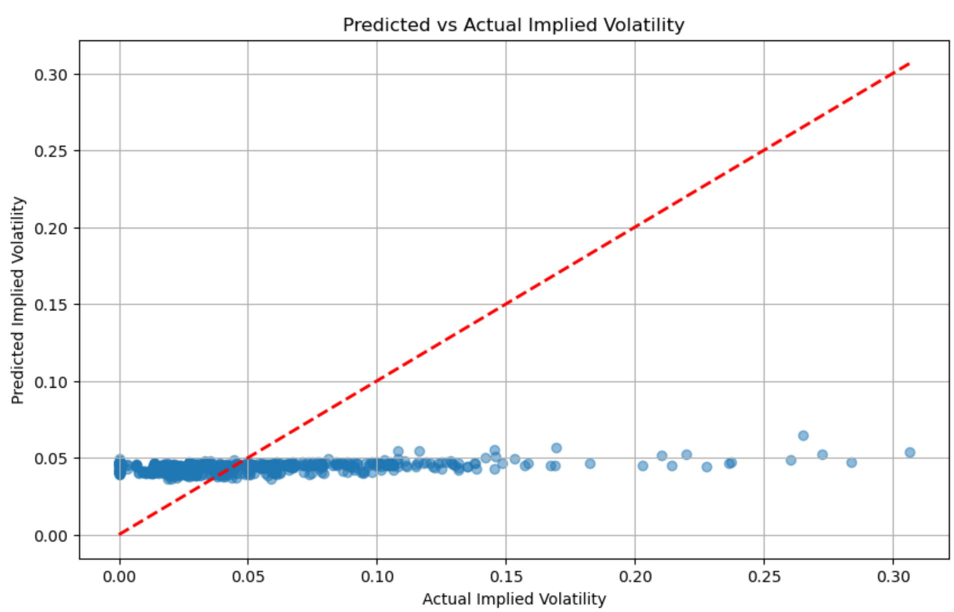
4.1 Learning Curves, MSE, RMSE

The model's performance on the test set is evaluated using Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). The MSE on average over 10 iterations was 0.021, and the RMSE is 0.145. These values suggest that the model captures the relationships between the input features and implied volatility, as evidenced by the low error metrics. The Training Loss Curve and Validation Loss Curve, plotted below, show the MSE loss per epoch. Both curves exhibit a consistent decrease and stabilize towards the end of training with a minimal gap between them. This behavior indicates that the model is learning effectively from the training data and generalizing well to unseen validation data. The convergence of the curves suggests that the model's parameters are well-optimized, reducing both training and validation losses over time.



4.2 Predicted Values vs Actual Values

Despite having a low MSE and RMSE, the model did not perform well in making accurate predictions. The scatter plot below, which compares the Predicted Values to the Actual Values of the test dataset, highlights this discrepancy.

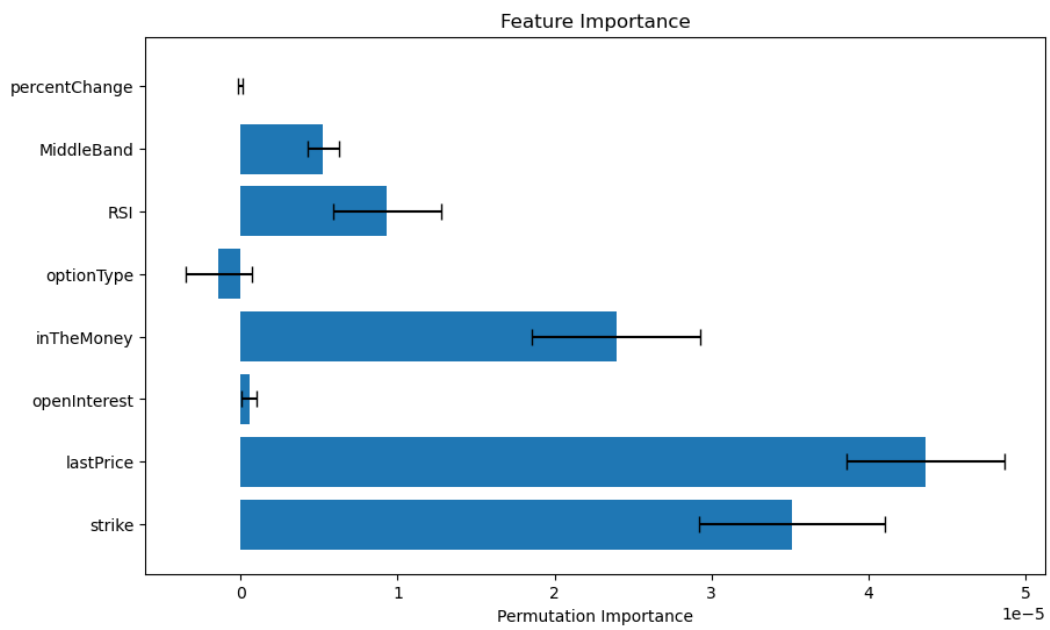


The plot, featuring 598 data points, reveals that the model's predictions are heavily clustered around 0.05, rather than aligning closely with the regression line. This clustering suggests that the model is likely underfitting the data, meaning it is too simplistic to capture the underlying complexity of the relationships between the features and implied volatility.

The low MSE and RMSE may be misleading due to the narrow range of implied volatility values in the dataset. This narrow range, coupled with a relatively small and potentially unrepresentative dataset, could contribute to the model's inability to generalize well. Additionally, the model may suffer from underfitting because it has too few layers relative to the number of features, limiting its capacity to learn and represent complex patterns in the data.

5 Future Work

As indicated by the visualizations, it is clear that the model requires substantial improvements. To enhance its performance, increasing the dataset size to over 10,000 rows may be more sufficient than 3,000 given the complexity of the problem and the number of input features. This may involve sourcing data from platforms other than Yahoo Finance. Additionally, a re-evaluation of the features used in the model is also necessary. The plot below illustrates feature importance based on permutation importance, which measures how much the model's performance (as indicated by MSE) deteriorates when each feature's values are randomly shuffled. This approach helps determine the significance of each feature in the model.



From this analysis, Last Price and Strike Price appear as the most important features, with the "In the Money" indicator also proving valuable. Consequently, these features should be retained in the dataset. On the other hand, features such as RSI, Middle Bollinger Band, Option Type, Percent Change, and Open Interest appear to have minimal impact on the model and can be considered for removal.

Further model improvements could involve reconsidering the architecture, possibly by adding more hidden layers to better capture complex relationships. Once the model is refined and capable of accurate predictions, the focus will shift to applying it to enhance options trading strategies.

6 Conclusion

In summary, this paper explores the use of a neural network to forecast implied volatility in stock options. While the model achieved low Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), indicating some predictive capability, it faced challenges with underfitting and limited accuracy. Future work will focus on expanding the dataset, re-evaluating feature importance, and enhancing the model's architecture to improve performance.