Machine Learning For Accurate Option Pricing

Yin Qiuhao, Jerome Lee Zong Huan, Joel Sng Zhen Wei, Eng T-Leng, Leong Yew Kit

School of Computing and Information Systems, Singapore Management University, 80 Stamford Rd, Singapore 178902, Singapore

Abstract. This report investigates the application of machine learning (ML) models in options pricing, comparing their performance against the traditional Black-Scholes model (BSM). BSM, a foundational framework in financial derivatives, offers interpretability and ease of use but struggles with real-world complexities due to its reliance on assumptions like constant volatility and a log-normal distribution of returns. By contrast, ML models such as Support Vector Regression (SVR), Artificial Neural Networks (ANN), and Long Short-Term Memory networks (LSTM) leverage data-driven methodologies to capture non-linear and dynamic relationships in financial markets.

Using SPX and SPXW options data spanning six trading days, this study evaluates the predictive capabilities of these models through metrics such as Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R^2). The results reveal that ML models consistently outperform BSM in terms of accuracy and flexibility, with the ANN model achieving the best performance for SPX call options and SPXW put options. However, ML models face challenges such as computational demands, sensitivity to data quality, and overfitting risks.

This study highlights the transformative potential of ML in financial modeling and discusses future opportunities to enhance robustness and expand applications to diverse market scenarios.

Keywords: Option Pricing · Neural Networks · Support Vector Regression · Machine Learning · Black-Scholes model

1 Introduction

Options, as a cornerstone of modern financial markets, are financial derivatives that grant their holders the right, but not the obligation, to buy (call) or sell (put) an underlying asset at a specified price on or before a predetermined expiration date. They are integral tools for hedging, speculation, and portfolio management, offering traders and institutions mechanisms to manage risk and exploit market opportunities. Options derive their value from a combination of factors, including the underlying asset's price, time to maturity, volatility, and prevailing market conditions. Their widespread adoption underscores their importance; in 2023 alone, over 3.7 billion contracts were traded on the Chicago Board Options Exchange, reflecting their critical role in financial decision-making and market dynamics [3].

A fundamental challenge in the options market is determining the accurate price of these derivatives. Mispricing can lead to significant financial risks, including poor investment decisions and ineffective hedging strategies. Over the decades, theoretical models have been developed to address this issue, with the Black-Scholes model (BSM) being one of the most influential. The BSM provides a closed-form solution for pricing European-style options by assuming that asset prices follow geometric Brownian motion, volatility and risk-free rates remain constant, and markets are frictionless. Despite its elegance and widespread application, the BSM is constrained by these assumptions, which often fail to capture real-world market behaviors such as volatility clustering, price jumps, and the volatility smile. Consequently, extensions to the BSM, as well as alternative models like the Heston model and Monte Carlo simulations, have emerged to address these gaps, albeit with varying degrees of success.

1.1 Motivation

As financial markets evolve, traditional models like the BSM increasingly face limitations in addressing the complexities of modern trading environments. Their inability to adapt to dynamic market conditions and incorporate diverse factors such as implied volatility, macroeconomic indicators, and market sentiment has necessitated the exploration of alternative methodologies. Machine learning (ML) has emerged as a promising approach in this domain. Unlike traditional models, ML relies on data-driven techniques to uncover patterns and relationships within historical data, eliminating the need for rigid assumptions. Techniques like Support Vector Regression (SVR), Artificial Neural Networks (ANNs), and Long Short-Term Memory (LSTM) networks offer the flexibility to model non-linear and dynamic relationships, adapt to market changes, and incorporate a broader range of inputs.

However, adopting ML in options pricing presents its challenges. While ML models often outperform traditional methods in predictive accuracy, they are frequently criticized for their "black box" nature, which limits interpretability and stakeholder trust. Moreover, they require vast amounts of high-quality data and computational resources, raising concerns about overfitting and robustness in the face of extreme market conditions. This report aims to investigate the theoretical and practical aspects of traditional and ML-based models for options pricing, comparing their strengths, limitations, and potential to address the demands of contemporary financial markets.

2 Literature Review

In our research spanning across multiple existing studies, we aimed to learn more about the processes and techniques used to predict option prices. In the following sections below, we will be focusing our attention more on models which we believe will deliver the most meaningful insights with respect to the scope of our studies.

2.1 Introduction to Baseline Model: Black-Scholes Formula

The Black-Scholes Model (BSM), developed by Fisher Black and Myron Scholes in 1973, is a mathematical framework for pricing options based on variables such as the underlying asset's price, strike price, time to maturity, volatility, and risk-free interest rate.

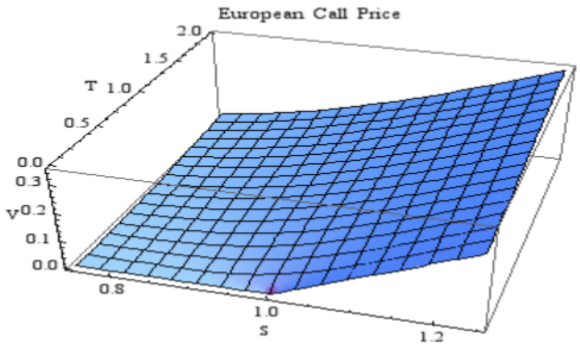

Fig. 1: European call valued using the Black-Scholes pricing equation

The model relies on several assumptions, including that stock prices follow a lognormal distribution, no dividends are paid, options can only be exercised at expiration (limiting its use to European options), and that the stock market operates in a random walk. It also assumes a frictionless market with no transaction costs and constant risk-free rates. The BSM is mathematically represented by a second-order partial differential equation, leading to formulas for pricing European call and put options. Assumptions and formulas will be further explained under Chapter 4 - Methodology

However, it has limitations, including its inapplicability to American options, dependence on rarely constant risk-free rates, and neglect of dividends or other returns typical in trading [20].

2.2 Supervised Machine Learning: Support Vector Machine (SVM)

Unlike the Black-Scholes framework, which relies on specific assumptions about market conditions, SVM can learn the relationships between key economic parameters—such as stock price, exercise price, historical volatility, maturity date, and interest rate—without these constraints, leading to effective pricing. A key advantage of SVM lies in its ability to find a unique solution by framing the problem as a convex optimization task. This ensures that the solution is not dependent on the initial conditions or local minima, a common challenge in other machine learning techniques.

Additionally, SVM maximizes the margin between the predicted outputs and the actual data, which enhances its generalization capability. By finding the optimal hyperplane in feature space, SVM balances the trade-off between accuracy on the training set and performance on unseen data. This margin maximization makes SVM particularly robust in capturing nonlinear relationships in option pricing.

Support Vector Machines are highly effective for classification tasks in option pricing. They can classify options into categories such as "in the money," "at the money," or "out of the money" based on key features like strike price, underlying asset price, volatility, and interest rates. By training on historical data, SVMs learn the relationships between these features and the option's position relative to market conditions. This classification capability aids traders in understanding market dynamics, optimizing trading strategies, and managing risk effectively. [13]

For regression tasks, SVMs are used to predict option prices or implied volatility by modeling the nonlinear relationships between input variables and target outcomes. These models leverage features such as asset prices, time to maturity, and volatility to provide accurate price estimates, enabling informed trading decisions. Adaptive SVMs further enhance predictive performance by dynamically adjusting parameters to reflect changing market conditions, making them particularly valuable in financial forecasting and navigating. [13]

Comparatively, other machine learning algorithms like random forest and multiple linear regression have also been evaluated in this context. Research indicates that random forest models can better capture the relationship between inputs and outputs than SVM, highlighting the diverse capabilities of machine learning techniques in pricing options. However, challenges remain regarding hyperparameter optimization and the potential for high volatility in trading strategies derived from these models, underscoring the need for careful analysis in applying these methods. [15].

2.3 Artificial Neural Networks (ANN)

Option pricing using deep learning (DL) represents an emerging field that leverages artificial neural networks (ANNs) to model the intricate dynamics of financial markets and accurately price derivatives such as options. In recent years, several frameworks have emerged to apply ANNs to the pricing of European options, as demonstrated by Funahashi [6], Liu et. al. [16], and Horvath et. al. [11].

A feedforward ANN operates by transmitting information in a unidirectional manner, and comprises input nodes, hidden nodes, and output nodes corresponding to a desired output of interest. ANNs have historically been influenced by the structure of the human brain and biological neural networks. Unlike simpler machine learning models, ANNs overcome linearity constraints by utilizing multiple layers of interconnected neurons [18]. Learning in ANNs is achieved through back-propagation, a process that modifies weights and biases using gradient descent.

In contrast to traditional methods, such as the Black-Scholes model (BSM), which depend on simplifying assumptions like constant volatility and log-normal returns, ANN and DL approaches are capable of capturing complex, nonlinear interactions between market variables that influence option prices. By training neural networks on historical market data, ANNs and DL models can effectively generalize to new market conditions, leading to improved prediction accuracy [20].

Moreover, comparing to conventional machine learning techniques such as support vector machines (SVM) and k-nearest neighbors (kNN), ANNs offers several distinct advantages. These include the ability to perform unsupervised feature learning, superior generalization capabilities, and efficient training on large datasets [5], making it particularly well-suited for complex financial applications.

Funahashi [6] proposed an ANN framework leveraging asymptotic expansion to improve computation efficiency, stability, and accuracy in option pricing. Horvath et al. [11] proposed an ANN framework to predict option pricing, integrating a fractional volatility model and utilizing a two-phase approach of approximation and calibration to enhance the performance of the model. Liu et al. [16] introduced an ANN-based system aimed at calibrating parameters for well-established financial pricing models, including the Heston and Bates frameworks. Spiegeleer et al. [4] developed a Bayesian non-parametric method using Gaussian process regression to train ANNs for American option pricing, leveraging carefully chosen kernel functions and efficient covariance matrix inversion to achieve significant training speed improvements. Despite these contributions, certain limitations remain, such as a lack of control over inversion functions in Horvath et al.'s method, substantial computational overhead for large datasets, issues like exploding or vanishing gradients in more intricate models, and model instability due to bell-shaped sensitivity of option prices to input parameters in Funahashi's method.

2.4 Recurrent Neural Networks (RNN): Long Short Term Memory (LSTM)

Long Short-Term Memory (LSTM) models are a type of recurrent neural network (RNN) designed to handle time-dependent data effectively, particularly in time-series forecasting. Unlike traditional neural networks, LSTMs introduce transition weights that allow the model to retain memory of previous states, enabling it to use historical information to predict future values. LSTMs consist of four key components: input gates, forget gates, cell states, and output gates, which together help mitigate the vanishing gradient problem that often limits the efficiency of simple RNNs. These show superior performance compared to classical time series models and traditional networks across various applications [20].

In a study made by researchers from Stanford University, they employed an LSTM approach to explore whether additional layers could learn features as informative as or more informative than historical volatility, which is a key component in the Black-Scholes and Multi Layer Perceptron (MLP) models. Despite

their efforts, the LSTM model underperformed compared to the MLP models, indicating that further refinement is needed. Moreover, they recognized that predicting equilibrium prices instead of bid/ask values could have contributed to the suboptimal performance, suggesting that outputting bid/ask prices might yield better results [14].

Furthermore, in a research article written by Habib Zouaoui and Meryem Nadjat Naas, they explored Deep Learning techniques using LSTM models for option pricing, comparing it to other models such as the BSM and Gated Recurrent Unit (GRU). From their findings, the LSTM model demonstrated superior pricing accuracy for options compared to the Black-Scholes Model (BSM) and GRU models, excelling in metrics like MAE, MSE, and RMSE. For call options, LSTM achieves 2.33%, 13.58%, and 3.69%, respectively, and for put options, 4.44%, 54.30%, and 7.37%. In contrast, the BSM shows the highest errors, making it the least reliable, while the GRU performs moderately but not as well as the LSTM [20].

Options Type	Model	Train/Test (%)	Epochs	Time	MAE	MSE	RMSE
Call	BSM	80/20	200	1s 3ms/step	2.84	16.39	4.05
	LSTM	80/20	200	0s 4ms/step	2.33	13.58	3.69
	GRU	80/20	200	2s 13ms/step	2.65	15.21	3.90
Put	BSM	80/20	200	0s 1ms/step	4.70	54.46	7.38
	LSTM	80/20	200	0s 6ms/step	4.44	54.30	7.37
	GRU	80/20	200	0s 4ms/step	4.60	55.16	7.43

Fig. 2: Model results from various Option Pricing Models

This confirmed their hypothesis which suggests that deep learning models offer more accurate and robust predictions for option pricing by effectively capturing the complex, dynamic relationships between an underlying asset's risk, uncertainty, and its option price. However, they also mentioned that DL models require substantial amounts of high-quality data, thorough validation, and careful interpretation. Additionally, their performance can vary depending on the specific problem and the characteristics of the data used.

3 Datasets

Building upon our previous project proposal, which utilised three datasets sourced from the Chicago Board Options Exchange [2], we have expanded our dataset to include three additional data files, resulting in a total of six datasets. These datasets consist of SPX and SPXW option prices recorded at 3600-second intervals across six distinct trading days. To enhance the reliability of our analysis, we have filtered the data to focus exclusively on SPX options, as SPXW options, with their weekly expirations, are more susceptible to speculative behaviour and

heightened random volatility. This section provides a comprehensive exploration of the expanded dataset, including an in-depth exploratory data analysis (EDA) of the newly incorporated data. In addition to validating previous findings, this analysis aims to uncover insights from attributes not examined in the prior iteration, further enriching the understanding of the data and supporting the development of more robust models.

3.1 Overview

In total, the dataset contained 1021314 rows of data for SPX, split evenly into 510657 rows of calls and 510657 rows of puts. The data was collected on trading Fridays spanning 4 months from July 2024 to October 2024. Close dates for the data include: 2024-07-26, 2024-09-27, 2024-08-30, 2024-10-11, 2024-10-18, 2024-10-25.

3.2 Distribution Analysis

The distribution of implied volatility reveals patterns such as the "volatility smile" or "skew," which indicate varying levels of risk perception at different strike prices. Analysing this distribution can help traders identify areas of heightened uncertainty or mispricing. From our analysis, it can be seen that calls have a wider distribution, with a greater positive skew, indicating more variation in perceived calls than puts (Figure 3).

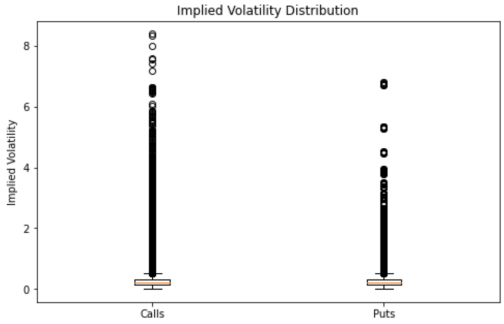

Fig. 3: Distribution of Implied Volatility between Calls and Puts

Option price distribution provides us insights into how the prices of call and put options are spread across various strike prices and expiration dates. This analysis often highlights pricing anomalies and clustering effects, such as the concentration of option activity near the money (ATM) strikes. From our analysis, we can see that calls have a stronger positive skew, suggesting more cheap call options being bought, suggesting that many options being bought are

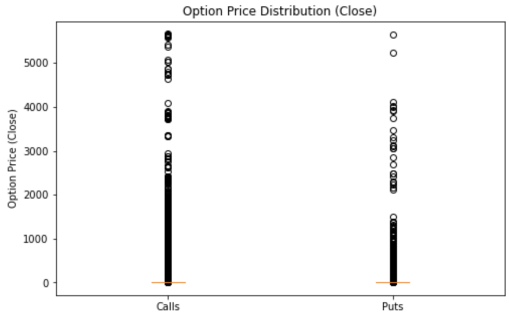

Fig. 4: Distribution of Option Price Distribution between Calls and Puts

currently out of the money and that traders are positioning for positive price movements (Figure 4).

The distribution of trade volume reflects market participation and liquidity at various strike prices and expirations. High volumes near key price levels can signal strong market interest or hedging activity, while sparse volumes may indicate illiquid conditions that could lead to wider bid-ask spreads. From our analysis, we can see that calls have a weaker positive skew. This suggests stable market sentiment, with limited speculative behaviour (Figure 5).

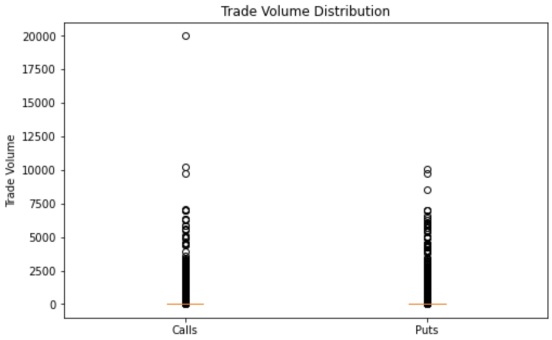

Fig. 5: Distribution of Trade Volume Distribution between Calls and Puts

Open interest measures the number of outstanding option contracts and provides a snapshot of market positioning. Analysing its distribution can highlight areas of significant activity, such as strike prices with large hedging or speculative positions. From our analysis, we can see that calls have a weaker positive skew. This suggests that there are alot of call trades occurring and at high volumes, indicating trader confidence in upwards price movements of the SPX (Figure 6).

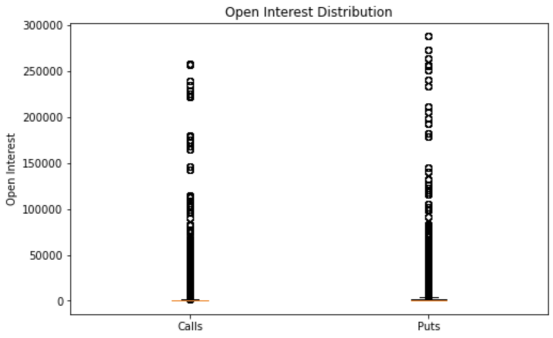

Fig. 6: Distribution of Open Interest Distribution between Calls and Puts

3.3 Correlation Analysis

Correlation analysis examines the relationships between various factors influencing option markets, such as price movements, volatility changes, and trading activity. For example, a strong correlation between IV and option prices might reflect heightened sensitivity to market sentiment, while weak correlations may suggest independent drivers or inefficiencies.

From our analysis of our numerical variables, we can see that some variables are implicitly related or derived from one another, and can be expected to have strong correlations as shown in Figure 7. A key note is that calls and puts both display opposite correlations for some variables, what might be positive correlation for puts may be negative for calls. This warrants training of separate models for each option type.

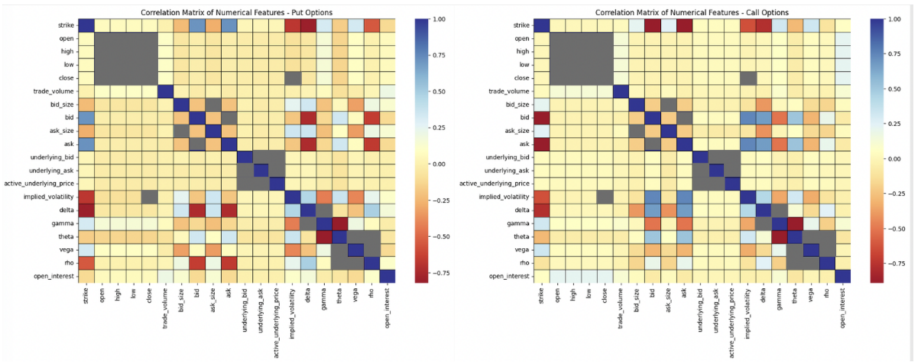

Fig. 7: Correlation Analysis of Numerical Features for Calls and Puts

3.4 Time Analysis

Time analysis in options markets focuses on the impact of temporal factors, such as time to expiration, intraday trading patterns, and seasonal effects, on key metrics like option prices, implied volatility, and trading activity. Understanding how these metrics evolve over time allows market participants to better manage time decay, also known as theta decay, and align their strategies with predictable patterns.

In our analysis, we included the new dates as can be seen in the figure below (Figure 8), and we observed roughly similar trends in price throughout the day. This suggests predictable asset behaviour, a stable market sentiment to this option, and roughly consistent volatility.

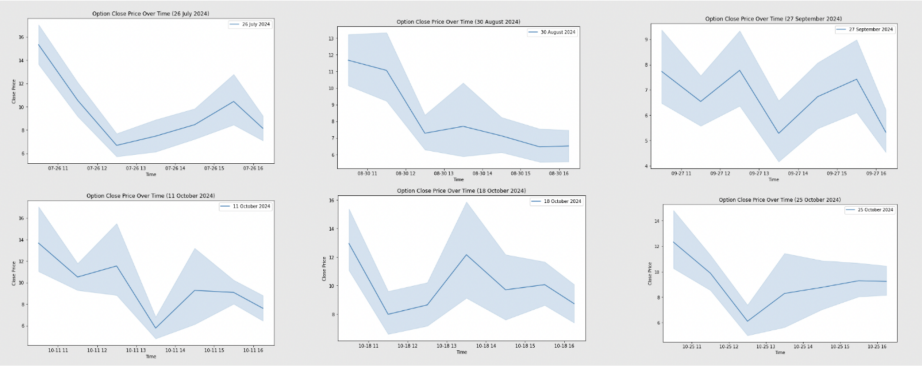

Fig. 8: Time Analysis of Option Close Price over Time

3.5 Greek Analysis

The Greeks are essential metrics in options trading that measure the sensitivity of an option's price to various factors, providing insights into risk and price behaviour. These include Delta, Theta, Gamma, Vega, and Rho, each derived from options pricing models like Black-Scholes. Traders use these metrics to assess how options respond to changes in the underlying asset's price, time to expiration, volatility, and interest rates [9].

Delta measures how much an option's price changes in response to a $1 movement in the underlying asset. For call options, delta ranges from 0 to 1, while for put options, it spans from 0 to -1. Delta also represents the probability of an option expiring in the money and helps traders calculate the hedge ratio for creating a delta-neutral position.

Gamma, on the other hand, indicates how delta changes with movements in the underlying asset, offering insights into delta stability. Gamma is highest

for at-the-money options and increases as expiration approaches, highlighting sensitivity to small price changes.

Theta quantifies an option's time decay, showing how much its value decreases as expiration nears. At-the-money options experience the highest time decay, particularly as they approach maturity. Long options typically have negative theta, while short options have positive theta.

Vega measures sensitivity to changes in implied volatility, with higher vega values indicating greater price changes in response to market volatility. Options with longer times to expiration and at-the-money status are most affected by changes in volatility.

Finally, **Rho** gauges sensitivity to interest rate changes, reflecting how a 1% interest rate shift affects an option's price. It has a more pronounced impact on longer-term options.

From our analysis of the Greeks in relation to the closing prices for both calls and puts (Figure 9), it is evident that these relationships are complex and highly non-linear. This complexity presents significant challenges for modelling and prediction, as the interactions between the Greeks and option prices are influenced by multiple dynamic factors, including volatility, time decay, and changes in the underlying asset's price.

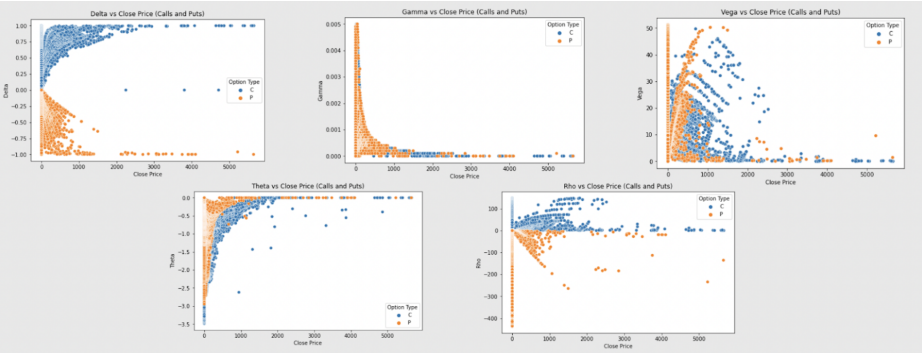

Fig. 9: Compilation of Correlation Analysis of Greeks

3.6 Feature Extraction

In addition to the variables mentioned above, we explored a variety of other relevant features that could impact option pricing, including historical volatility, trading volume, and open interest. This involved identifying and transforming raw data into meaningful variables that can be used for model training and analysis. The new features we extracted are as follows.

1. **Time to Maturity:** In order to obtain time to maturity, the maturity of each option has to be obtained, which was calculated by determining the

number of days between the expiration date (specified in the "Expiration" variable) and the trading date of the option. This maturity will then be divided by 252, representing the standard number of trading days in a year, to standardise the metric and express it as a proportion of a year, thus obtaining time to maturity in years.
2. **Risk-Free Rates:** Risk-free rates were derived from government bond rates published by the U.S. Department of the Treasury. These rates were matched to the calculated time to maturity for each option to ensure accuracy and consistency in representing the risk-free rate used in pricing models.
3. **Option Price (Bid-Ask Spread):** The option price was determined by calculating the midpoint of the bid and ask prices, leveraging the "Bid" and "Ask" variables. This approach helps to mitigate the impact of bid-ask spreads and provides a more representative estimate of the option's fair value in the market.
4. **Moneyness:** Moneyness, a measure of an option's in-the-money or out-of-the-money status, was calculated as the ratio of the underlying asset's price (S) to the option's strike price (K). This metric provides a clear indicator of the relative position of the underlying asset's price to the option's exercise price.

4 Methodology

4.1 Black-Scholes model

The Black-Scholes model, introduced in 1973 by Fisher Black and Myron Scholes, revolutionized the field of financial derivatives by providing a robust mathematical framework for pricing options [1]. Their model posits that the value of an option is derived from specific variables: the current price of the underlying asset, the strike price, the option's time to maturity, the asset's volatility, and the risk-free interest rate. This section elaborates on the assumptions underlying the model, the mathematical formulation, and its application as a baseline for this study.

4.1.1 Assumptions of the Black-Scholes Model

The Black-Scholes-Merton (BSM) model operates under a set of idealized assumptions [17]:

1. **Lognormal Distribution:** Asset prices are assumed to follow a lognormal distribution, ensuring non-negative values bounded by zero.
2. **No Dividends:** The model does not account for dividend payouts during the option's life.
3. **European Options:** It assumes that options are exercised only at maturity, making it most applicable to European-style options.
4. **Random Walk:** Stock prices are assumed to follow a stochastic process with unpredictable market direction.

5. **Frictionless Market:** Transaction costs, such as commissions or brokerage fees, are assumed to be absent.
6. **Constant Risk-Free Interest Rate:** Interest rates are constant over time, reflecting a risk-free borrowing or lending environment.
7. **Normal Distribution of Returns:** Market volatility is assumed to be stable, with returns normally distributed.
8. **No Arbitrage:** The model assumes no risk-free profit opportunities, ensuring equilibrium pricing [12].

These assumptions, while idealized, provide a tractable framework for pricing options analytically.

4.1.2 Black-Scholes Formula

The BSM model is represented by a second-order partial differential equation [1]:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial^2 V}{\partial S} - rV = 0 \tag{1}$$

A key insight behind the equation is the concept of risk-neutral pricing, where the option's value can be hedged perfectly by dynamically buying or selling the underlying asset and a risk-free bond. This ensures that the option has a unique "fair" price, calculated by solving the equation under appropriate boundary and terminal conditions [17]. For European call and put options, the closed-form solutions are given as:

$$C(S_t, t) = N(d_1)S_t - N(d_2)Ke^{-r(T-t)} \tag{2}$$

$$P(S_t, t) = N(-d_2)Ke^{-r(T-t)} - N(-d_1)S_t \tag{3}$$

Where:

$$d_1 = \frac{1}{\sigma\sqrt{T-t}}\left[\ln\left(\frac{S_t}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)\right] \tag{4}$$

$$d_2 = d_1 - \sigma\sqrt{T-t} \tag{5}$$

And Table 1 below shows the description of symbols used in BSM.

These formulas were applied to the dataset to compute option prices as a baseline model. The results, shown in the comparison chart below, demonstrate reasonable accuracy relative to the observed market prices.

4.1.3 Results

The following performance metrics were obtained for the Black-Scholes model:

- **Root Mean Square Error (RMSE): 112.55**, indicating an average deviation of 112.55 units between predicted and actual option prices.
- **Mean Absolute Error (MAE): 60.69**, reflecting the average absolute prediction error in dollar terms.

Symbol	Description
Black-Scholes Model Symbols	
S_t	Current price of the underlying asset
K	Strike price of the option
$T - t$	Time to maturity (in years)
r	Risk-free interest rate
σ	Volatility of the asset's returns
$C(S_t, t)$	Price of a European call option
$P(S_t, t)$	Price of a European put option
$N(\cdot)$	Cumulative distribution function of the standard normal distribution
d_1	Intermediate calculation term for the Black-Scholes formula
d_2	Intermediate calculation term for the Black-Scholes formula
V	Option price as a function of time and stock price
$\frac{\partial V}{\partial t}$	Partial derivative of V with respect to time
$\frac{\partial^2 V}{\partial S^2}$	Second partial derivative of V with respect to stock price

Table 1: Description of Symbols Used in Black-Scholes Formulas

option_price	BS_calculated_option_price
5250.0	5255.042910125770
5245.9	5249.477910125770
5269.65	5273.577910125770
5284.75	5288.172910125770
5258.9	5263.0129101257700
5261.5	5265.367910125770
5258.4	5266.837910125770
0.025	0.0
0.025	0.0

Fig. 10: Acutal Price & BSM Calculated Price

– **R-Squared (R^2): 99%**, signifying that the model accounts for 99% of the variance in option prices.

These results illustrate the analytical precision of the Black-Scholes model, making it a strong baseline for comparison with machine learning methods. However, limitations such as sensitivity to assumptions and deviations in real-world market conditions underscore the need for more adaptive approaches.

4.2 Support Vector Regression Model

Support Vector Regression (SVR) model is an adaptation of Support Vector Machines (SVM) for regression tasks. SVR aims to find a function that approximates the relationship between input features and a continuous target variable while minimising prediction error, rather than classifying data into categories.

We used Scikit-learn's SVR to predict option prices and compare it to the Black Scholes model and other machine learning methods covered later.

4.2.1 Features Used for Training

The following features were selected for the input features:

1. Active Underlying Price (S): Current price of the underlying asset.
2. Strike (K): Exercise price of the option.
3. Time to Maturity (T): Remaining time in years before the option expires.
4. Implied Volatility (σ): Market-expected volatility of the underlying asset.
5. Risk-Free Rate (r): Interest rate for risk-free assets.

With Option Price as the target variable.

4.2.2 Model Details

We decided to test our models on calls and puts separately, hence the models are tested SPX calls. SPX puts, SPXW calls and SPXW puts. Before testing our model, a 80:20 train test split is done on each data set where the model is trained using the training data and tested on the testing data to account for overfitting. The data is also scaled so that all variables have a comparable scale which will improve model performance.

Radial Basis Function (RBF) Kernel was used for our SVR model due to the non-linear relationship data relationships and the ability to control the model's complexity through parameters such as the regularisation parameter (C) and gamma (γ) which provides robust performance against overfitting.

In order to achieve the best Bias-Variance Trade off, the right γ and C has to be used in the model. We utilised Scikit-learn's GridSearchCV to conduct hyperparameter tuning. The metric used was negative Root Mean Squared Error (RMSE) where RMSE will indicate the magnitude of prediction errors. GridSearchCV will find the best model for the right γ and C which will have the lowest RMSE (in this case for negative RMSE scoring, we need to multiply by -1).

We also tested using epsilon of 0.01 or 0.1 on models (default value for SVR model) and found that there was a negligible difference and decided to go with the default value of 0.01.

The final parameters are:

Parameter	**Value**
Regularisation (C)	1000
Gamma (γ)	"scale"
Epsilon	0.1 (default value)

Table 2: Final parameters used for the SVR model

We chose not to test C above 1000 as using 1000 provided a good RMSE and going over might overfit the training data. Additionally, using 1000 resulted in long training times in our models and over 1000 (e.g. 10000) will result in increased computational time. For γ, "auto" and "scale" were tested as these selections allow gamma to adjust based on data.

4.2.3 Results

We evaluated the performance of the models based on 3 metrics, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and coefficient of determination (R-square). Below are the results of the SVR models.

Metric	**SPX Calls**	**SPX Puts**	**SPXW Calls**	**SPXW Puts**
RMSE	17.20	44.03	17.06	13.37
MAE	4.55	4.58	6.79	5.29
R^2	0.99	0.99	0.99	0.99

Table 3: Evaluation metrics for SPX and SPXW calls and puts.

The SVR model has performed better than the Black Scholes model for both RMSE and MAE. A R-square of 0.99 also shows the model has fit the data and independent variables explain 99% of the variance in the dependent variable.

4.3 Recurrent neural networks (RNN)

Recurrent Neural Networks (RNNs) differ from traditional feedforward neural networks by introducing transition weights that enable information transfer across time steps [7]. These transition weights create a dependency between subsequent states, allowing the model to retain "memory" of past inputs. In RNNs, the hidden layers act as internal storage for information from earlier time steps, enabling the model to learn sequential patterns. The term "recurrent" signifies that the model applies the same function to each sequence element, using previous states to predict future outcomes [10].

4.3.1 Long short term memory (LSTM) model

The Long Short-Term Memory (LSTM) network, a gated memory unit, addresses the vanishing gradient problem inherent in standard RNNs [19]. Unlike simple RNNs, LSTMs incorporate a sophisticated gating mechanism that selectively retains or discards information. This mechanism is implemented through four key components: the input gate, forget gate, cell state, and output gate (Figure 11). These gates enable LSTMs to effectively capture long-term dependencies in data.

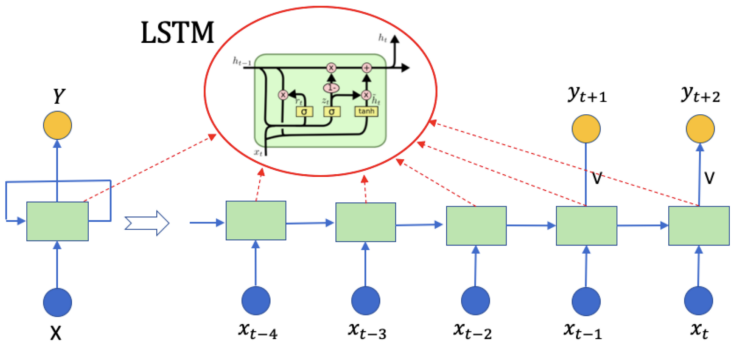

Fig. 11: LSTM structure

The input gate determines the relevance of new input data to the cell state. It uses a sigmoid activation function to scale values between 0 (irrelevant) and 1 (relevant). The forget gate determines which parts of the cell state should be discarded. This decision is crucial for filtering out irrelevant or outdated information. The cell state represents the internal memory of the LSTM, updated at each time step based on the contributions of the input and forget gates. The cell state retains information selectively over long sequences, ensuring the gradient flow is preserved [8]. The output gate determines which part of the cell state contributes to the hidden state, which in turn is passed to the next time step or used for predictions.

4.3.2 Features Used for Training

The following features were selected for training the LSTM model:

1. **Moneyness:** moneyness=active underlying price/strike
2. **Active Underlying Price (S):** Current price of the underlying asset.
3. **Time to Maturity (T):** Remaining time in years before the option expires.
4. **Implied Volatility (σ):** Market-expected volatility of the underlying asset.
5. **Risk-Free Rate (r):** Interest rate for risk-free assets.
6. **Option Type:** Encoded as numerical values for Call or Put options.

Given that the dataset contains six weeks of Friday trading data at 30-90 minute intervals, a sequence length of 5 was selected to capture sufficient temporal patterns without overloading the model.

4.3.3 Model Architecture

The LSTM model architecture, illustrated in Figure 12, comprises the following layers:

– First LSTM Layer: Contains 64 LSTM units to capture temporal dependencies.

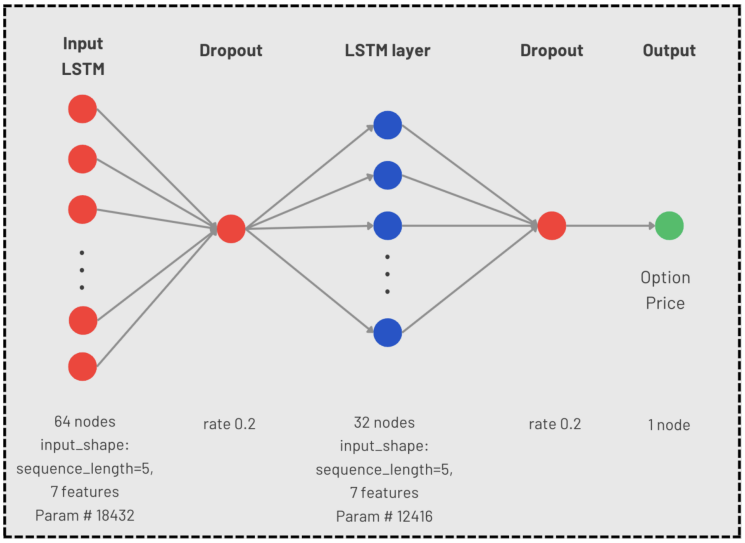

Fig. 12: LSTM model architecture

- Dropout Layer (Rate 0.2): Prevents overfitting by randomly deactivating units during training.
- Second LSTM Layer: Contains 32 units for further abstraction of patterns.
- Dropout Layer (Rate 0.2): Further reduces the risk of overfitting.
- Output Layer: Provides the final predictions based on the learned patterns.

This architecture balances complexity and efficiency, enabling the model to effectively analyze sequential data without overfitting.

4.3.4 Results

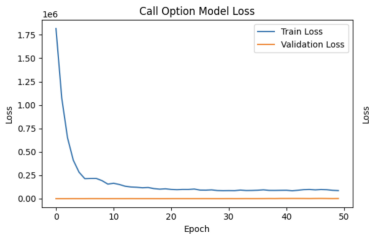

Fig. 13: Call option model training

Figures 13 and 14 display the training and validation loss curves for the call and put option models, respectively. The **call option model** demonstrates

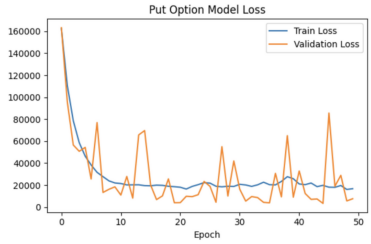

Fig. 14: Put option model training

excellent performance, with validation loss approaching near zero, indicating effective learning and minimal overfitting. In contrast, the **put option model** exhibits inconsistent validation loss. This irregularity suggests potential challenges in learning patterns, likely due to the smaller number of observations for put option trades in the dataset.

After training both models for 50 epochs, the following performance metrics were obtained:

Metric	**Call Options**	**Put Options**
Root Mean Square Error (RMSE)	87.273	40.803
Mean Absolute Error (MAE)	41.859	37.55
R-Squared (R^2)	96%	78.6%

Table 4: Evaluation metrics for Call and Put Options.

As shown in table 4, R-squre of 96% for call options, indicating that the model explains 96% of the variance in call option prices. And 78.6% for put options, suggesting moderate success in capturing the variance in put option prices.

These results indicate that the LSTM model outperformed the Black-Scholes model in predicting option prices, particularly for call options. However, its performance did not match the accuracy of the Artificial Neural Network (ANN) model discussed later.

A key limitation in leveraging the full potential of LSTMs lies in the dataset size and sequence length. With data spanning only six Fridays of trading and a sequence length of 5, the model's capacity to capture long-term dependencies is underutilized. LSTMs excel with richer datasets and longer sequences, where temporal dependencies are more prominent. Furthermore, sourcing granular option price data posed a significant challenge due to cost constraints, limiting the depth of our analysis.

In summary, while LSTMs showed promising results and surpassed the baseline Black-Scholes model, the data constraints hindered its performance, motivating our exploration of ANN as an alternative.

4.4 Aritificial Neural Network (ANN)

The final model employed by our group is the Artificial Neural Network (ANN), a well-established deep learning model. First inspired by the structure of biological neural networks and the functioning of the human brain, ANNs were introduced as a flexible approach to model complex patterns and non-linear relationships in data [18].

A feedforward ANN is composed of three primary components: an input layer that represents the features of the data, one or more hidden layers that process information through interconnected neurons, and an output layer that delivers the predicted outcome. The architecture's adaptability stems from its ability to modify weights and biases during training via back-propagation, a gradient-based optimization technique.

ANNs are particularly advantageous in tasks that require learning from large datasets and capturing intricate, nonlinear dynamics, such as option pricing and other financial modeling tasks.

4.4.1 Initial Approach: Single-Stage ANN for Options Pricing

Initially, our group employed an Artificial Neural Network (ANN) model, which we refer to as the 'Single-Stage ANN', in contrast to our final model. The architecture of this model consisted of an input layer with a number of nodes equal to the number of input features, followed by a first hidden layer with 64 nodes, a second hidden layer with 32 nodes, and an output layer with a single node for predicting the option price (Figure 15).

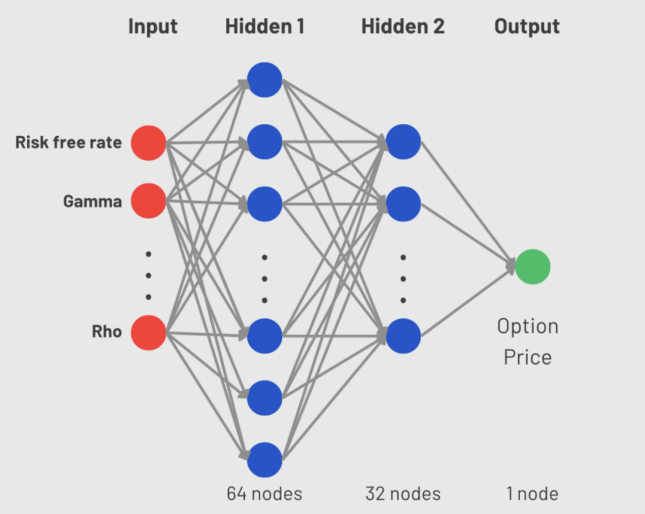

Fig. 15: Initial Single-Stage ANN Architecture

One of the main challenges encountered with this architecture stemmed from the nature of our dataset, which consisted of a callbook of options data. This dataset documents all available options at specific points during the trading day, including those that were not traded. For options that were traded, the dataset recorded the corresponding closing price, while for untraded options, a value of zero was recorded as the closing price. The abundance of zero values, due to the non purchased options, introduced a significant amount of noise into the data. This not only skewed the model's performance but also led to an overrepresentation of zero values, which negatively impacted the model's ability to learn meaningful patterns and generalise effectively.

This negative performance is reflected in the model's results, which exhibited low RMSE but a high degree of overfitting to our data as evidenced by the model's training and validation loss (Figure 16).

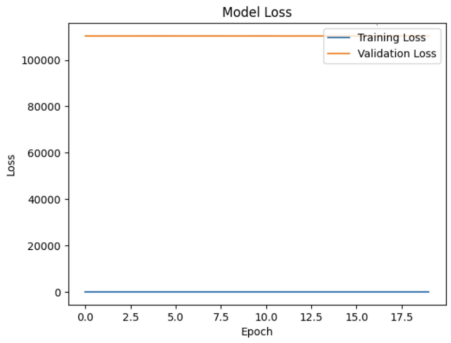

Fig. 16: Single-Stage ANN Model Training

4.4.2 Refinement to Two-Stage ANN Model

In response to the limitations observed with our initial single-stage ANN, our group carefully analyzed the challenges and designed a more robust solution in the form of a two-stage ANN model. The aim of this refinement was to address the overfitting issue and improve the model's ability to generalize, as well as to handle the specific characteristics of our data more effectively.

Stage 1: Binary Classification for 'Buyable' Options

The first stage of the refined model focuses on a binary classification task, where the model is trained to predict whether an option is likely to be 'buyable'. In the context of our data, an option is considered 'buyable' if it has a significant probability of being purchased. This stage helps filter out options that are unlikely to be traded, based on historical data and certain features that may indicate whether an option is typically exercised.

By classifying options as either 'buyable' or 'not buyable', this stage serves to pre-process the data by eliminating options with no real trading potential (those with a closing price of zero, for example). Only rows of data that are identified as 'buyable' move forward to the second stage for price prediction, ensuring that the model focuses only on relevant data and avoids training on unpurchased options.

Stage 2: Price Prediction for 'Buyable' Options

In the second stage, the ANN model conducts a regression task specifically for the 'buyable' options identified in Stage 1. The goal of this stage is to predict the price of each option that is classified as likely to be exercised. This involves taking the features of the options that passed through the first stage and using them to predict the continuous value of the option's price.

By separating the classification and regression tasks into two distinct stages, the model is able to learn more effectively, with the first stage helping to filter the data and the second stage focusing solely on price prediction for options that are most relevant. This approach mitigates the problem of overfitting to irrelevant data, as the training process is now more focused and efficient.

4.4.3 Features used

The following features were selected for training the ANN models:

1. **Strike**: Exercise price of the option.
2. **Trade Volume**: Total volume of the option contracts traded during the specified time.
3. **Active Underlying Price**: Current price of the underlying asset.
4. **Close**: The final traded price for that day of trading.
5. **Implied Volatility**: Market-expected volatility of the underlying asset, reflecting uncertainty or risk.
6. **Delta**: Rate of change of the option's price with respect to the underlying asset's price.
7. **Gamma**: Rate of change of delta with respect to changes in the underlying asset's price, representing convexity.
8. **Theta**: Rate of change of the option's price with respect to the passage of time (time decay).
9. **Vega**: Rate of change of the option's price with respect to changes in implied volatility.
10. **Rho**: Rate of change of the option's price with respect to changes in the risk-free interest rate.
11. **Open Interest**: Total number of outstanding option contracts that have not been exercised or closed.
12. **Risk-Free Rate**: Interest rate for risk-free assets, such as government bonds.
13. **Time to Expiration**: Time remaining in years until the option expires, derived from the expiration date.
14. **Bid-Ask Spread**: Difference between the bid price and the ask price, derived from the bid and ask columns.

4.4.4 Model architecture

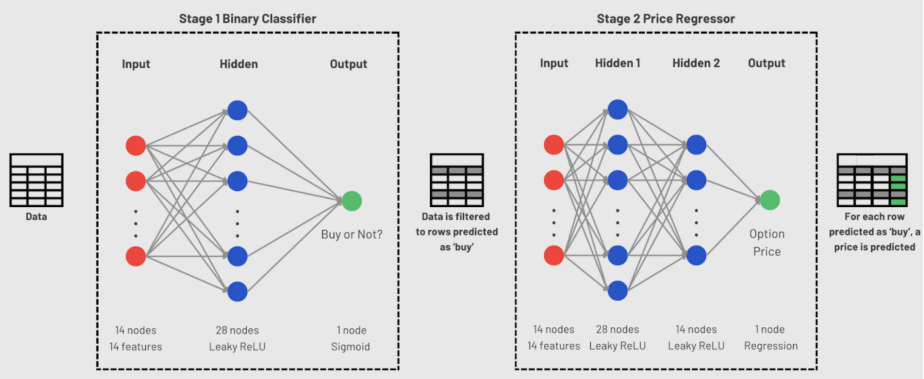

Fig. 17: Final Two-Stage ANN Architecture

The ANN model, shown in Figure 17, consists of two sub-models: a Binary Classifier ANN and a Price Regressor ANN. The architecture is tuned using scikit-learn's RandomSearchCV for hyperparameter optimization, adjusting the number of units in each hidden layer for different options types (SPX Puts, SPX Calls, SPXW Puts, and SPXW Calls). The following is a general framework our models follow, in our code hyper parameters of hidden layer units, model learning rate, batch size, and epochs vary between models (SPX Puts, SPXW Calls, SPXW Puts, SPXW Calls) due to hyper parameter tuning.

Binary Classifier ANN:

- First Input Layer: Contains 14 units matching 14 relevant features in data.
- Hidden Layer 1: Contains between 14 to 56 units with leaky ReLU function to learn relationships between features.
- Output Layer: 1 unit with a sigmoid function to predict option 'buyability'.

Price Regressor ANN:

- First Input Layer: Contains 14 units matching 14 relevant features in data.
- Hidden Layer 1: Contains between 14 to 56 units with leaky ReLU function to learn relationships between features.
- Hidden Layer 2: Contains between 7 to 28 units with leaky ReLU function for further abstraction between patterns.
- Output Layer: 1 unit for final price prediction.

This architecture optimally balances complexity and efficiency. The adjusted number of units via hyperparameter tuning allows the model to capture intricate patterns without overfitting, ensuring accurate price predictions and effective classification.

Our group recognised the potential of this model and broke down its formulation into the forwards propagation formulas and backwards propagation formulas used by the sk learn python package for each stage ANN to deepen understanding of it.

Forwards Propagation Formulas (Binary Classifier)

$$z_{n-1} = a_{n-2}W_{n-1} + b_{n-1} \tag{1}$$

$$a_{n-1} = \text{LeakyReLU}(z_{n-1}) \tag{2}$$

$$z_n = a_{n-1}W_n + b_n \tag{3}$$

$$a_n = \sigma(z_n) \tag{4}$$

$$\hat{y} = a_n \tag{5}$$

Backwards Propagation Formulas (Binary Classifier)

$$\delta_n = \frac{\partial \mathcal{L}}{\partial z_n - 1} = \hat{y} - y \tag{B-1}$$

$$\delta_{n-1} = \delta_n W_{n-1}^{\top} \cdot \text{LeakyReLU}'(z_{n-2}) \tag{B-2}$$

$$\frac{\partial \mathcal{L}}{\partial W_{n-1}} = a_{n-2}^{\top}\delta_n \tag{B-3}$$

$$\frac{\partial \mathcal{L}}{\partial b_{n-1}} = \delta_n \tag{B-4}$$

Forwards Propagation Formulas (Regressor)

$$z_{n-1} = a_{n-2}W_{n-1} + b_{n-1} \tag{1}$$

$$a_{n-1} = \text{LeakyReLU}(z_{n-1}) \tag{2}$$

$$z_{n-2} = a_{n-1}W_{n-2} + b_{n-2} \tag{3}$$

$$a_{n-2} = \text{LeakyReLU}(z_{n-2}) \tag{4}$$

$$z_n = a_{n-2}W_n + b_n \tag{5}$$

$$\hat{y} = z_n \tag{6}$$

Backwards Propagation Formulas (Regressor)

$$\delta_n = \frac{\partial \mathcal{L}}{\partial z_n} = \hat{y} - y \tag{B-1}$$

$$\delta_{n-1} = \delta_n W_n^{\top} \cdot \text{LeakyReLU}'(z_{n-1}) \tag{B-2}$$

$$\delta_{n-2} = \delta_{n-1} W_{n-1}^{\top} \cdot \text{LeakyReLU}'(z_{n-2}) \tag{B-3}$$

$$\frac{\partial \mathcal{L}}{\partial W_{n-1}} = a_{n-2}^{\top}\delta_{n-1} \tag{B-4}$$

$$\frac{\partial \mathcal{L}}{\partial b_{n-1}} = \delta_{n-1} \tag{B-5}$$

$$\frac{\partial \mathcal{L}}{\partial W_{n-2}} = a_{n-3}^{\top} \delta_{n-2} \tag{B-6}$$

$$\frac{\partial \mathcal{L}}{\partial b_{n-2}} = \delta_{n-2} \tag{B-7}$$

As per our module content, model weights and biases would be updated

$$W_i = W_i - \eta \cdot \frac{\partial \mathcal{L}}{\partial W_i} \tag{B-8}$$

$$b_i = b_i - \eta \cdot \frac{\partial \mathcal{L}}{\partial b_i} \tag{B-9}$$

4.4.5 Results

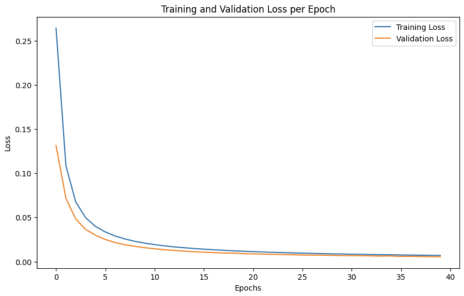

Fig. 18: Binary Classifier Training for SPX Calls

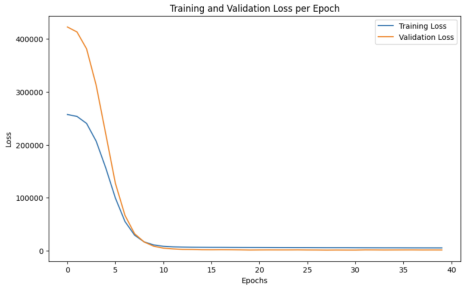

Fig. 19: Regressor Training for SPX Calls

SPXW Metric	**Calls**	**Puts**
Binary Classifier		
Accuracy	99%	99%
Precision	99%	99%
Recall	94%	99%
F1-Score	97%	99%
Price Regressor		
Root Mean Square Error (RMSE)	19.28	6.14
Mean Absolute Error (MAE)	6.14	3.14
R-Squared (R^2)	91%	97%

Table 5: Evaluation metrics for SPXW options: Calls and Puts.

SPX Metric	**Calls**	**Puts**
Binary Classifier		
Accuracy	99%	99%
Precision	99%	99%
Recall	97%	99%
F1-Score	99%	99%
Price Regressor		
Root Mean Square Error (RMSE)	3.82	20.04
Mean Absolute Error (MAE)	2.95	8.02
R-Squared (R^2)	99%	92%

Table 6: Evaluation metrics for SPX options: Calls and Puts.

5 Result & Discussion

In this section, we aim to evaluate the performance of machine learning models against the Black-Scholes model in predicting option prices. This assessment will focus on two key aspects: the accuracy of predictions and the model's goodness of fit, determining how well the model captures the overall trends within the data.

To do this, we will be utilizing the following statistical metrics:

1. **Root-Mean-Squared Error (RMSE)**: Measures the average magnitude of prediction errors, giving more weight to larger deviations. Lower RMSE values indicate more accurate predictions.
2. **Mean-Absolute Error (MAE)**: Represents the average absolute difference between predicted and actual values, providing a straightforward measure of prediction accuracy.
3. **R-squared ($\mathbf{R^2}$)**: Evaluates how well the model explains the variance in the data, with values closer to 1 indicating a better fit and greater predictive power.

5.1 Consolidated Results

For SPX Call Options, our proposed ANN model consistently outperforms other models, achieving superior results across all key metrics. Specifically, the ANN model demonstrates better predictive accuracy and fit, with RMSE, MAE, and R^2 scores of 3.82, 2.95, and 0.99, respectively.

For SPX Put Options, the SVR model emerges as the best-performing model, particularly excelling in MAE and R^2 metrics, with scores of 4.58 and 0.99, respectively, surpassing the performance of the ANN model in these areas.

Models	Call Options (SPX)			Put Options (SPX)		
	RMSE	MAE	Rsquare	RMSE	MAE	Rsquare
BSM	112.55	60.69	0.99	112.55	60.99	0.99
SVR	17.20	4.55	0.99	44.03	4.58	0.99
ANN	3.82	2.95	0.99	20.04	8.02	0.92
LSTM	40.803	37.445	0.786	87.273	41.859	0.964

Fig. 20: Comparison of Evaluation Metrics for Proposed Models for SPX Calls and Puts

For SPXW Call Options, the distribution is not as clear, with our proposed SVR model narrowly outperforming our ANN model, achieving slightly better results in RMSE and R^2 with scores of 17.06 (as compared to 19.28) and 0.99 (as compared to 0.91) respectively, while ANN performed better in MAE with a score of 6.14 (as compared to 6.79) For SPXW Put Options, the ANN model outperforms the SVR model particularly in RMSE and MAE, with scores of 6.14 and 3.14 respectively. However, SVR performed better in R^2 with a score of 0.99.

Models	Call Options (SPXW)			Put Options (SPXW)		
	RMSE	MAE	Rsquare	RMSE	MAE	Rsquare
BSM	112.55	60.69	0.99	112.55	60.99	0.99
SVR	17.06	6.79	0.99	13.37	5.29	0.99
ANN	19.28	6.14	0.91	6.14	3.14	0.97
LSTM	40.803	37.445	0.786	87.273	41.859	0.964

Fig. 21: Comparison of Evaluation Metrics for Proposed Models for SPXW Calls and Puts

5.2 Evaluation of Models

Overall, our analysis revealed that the ANN model performs best for SPX Call options and SPXW Put options, while the SVR model shows superior performance for SPX Put options and SPXW Call options. Despite achieving high R-squared values of 0.99, the Black-Scholes Model (BSM) consistently recorded the highest RMSEs and MAEs across all categories, indicating that while it explains the variance in the data reasonably well, it struggles with accuracy in absolute error terms compared to the machine learning models (Figure 22).

	SPX Call	SPX Put	SPXW Call	SPXW Put
Best Performance	ANN	SVR	SVR	ANN
Worst Performance	BSM	BSM	BSM	BSM

Fig. 22: Overall Evaluation of of Models based on Performance

6 Conclusion & Future work

Conclusion

The results of this study imply a great potential for machine learning models to be transformative in handling options pricing. Traditional models, such as the Black-Scholes Model, while serving as a strong foundation, are ultimately outdated, and limited by their reliance on assumptions such as constant volatility of over the lifespan of the option and a log-normal distribution of returns, which are impractical to real-life market conditions. These assumptions severely limit the Black-Scholes Model's ability to capture and interpret the complexities present in financial markets.

The machine learning models we utilised– SVR, ANN and LSTM, have demonstrated a superior predictive power to BSM. However, despite their superior accuracy in pricing options, machine learning comes with its own set of considerations. In contrast to BSM being a ready-made formula, machine learning models require a significant time investment. Varying complexities of models and sizes of training datasets impacts the amount of time and computational power required to train and utilise the model. Machine learning models are also heavily dependent on the quality of training data, and models trained on historical data may be susceptible to overfitting, and may struggle to generalise to extreme market conditions.

Future Work

Possibilities for future work our group has identified aims to enhance the applicability and performance of these machine learning models by addressing diversity of training data, improving feature engineering, and extending the coverage of option styles.

Our current models have been trained on datasets that reflect a stable and cautiously bullish market sentiment. While effective within this context, their generalisability to novel market conditions is unproven. To address this, our model can be trained on a broader range of market scenarios, such as bull markets with heightened positive sentiment (e.g. markets in the week after Trump's 2024 elections) or bear markets characterized by strong negative sentiment (e.g. Covid-19 pandemic trading conditions). This would ensure that the models are robust, generalisable, and adaptable to other market conditions.

More advanced feature engineering represents a possible area for further development. Incorporating additional macroeconomic indicators, such as unemployment rates, currency exchange rates, momentum indicators (e.g., MACD), the Volatility Index (VIX), volatility skew, and commodity prices, could allow the models to capture nuanced relationships and broader market dynamics. These enhancements could possibly improve predictive accuracy and enable the models to perform more effectively across diverse financial environments.

Finally, expanding the application of these models to other option types, such as American and Asian options, would broaden our models' utility. American options provide flexibility in exercise timing, while Asian options, with payoffs based on the average price of the underlying asset, suit assets with stable or predictable price paths. Future work could involve comparative studies with traditional methods, such as binomial and trinomial tree models, to further validate the performance of the machine learning approach and highlight its advantages over conventional techniques.

By addressing these areas, our models have the potential to offer a comprehensive, robust, and adaptable framework to effectively predict option prices in dynamic and evolving financial markets.

7 Acknowledgments

We would like to express our sincere gratitude to Professor Wang Zhaoxia for her guidance and support throughout the IS460 Machine Learning & Algorithm class.

References

1. Black, F., Scholes, M.: The pricing of options and corporate liabilities. Journal of Political Economy **81**(3), 637–654 (1973)
2. Cboe Data Services: Cboe data services. https://www.cboe.com/data, accessed: November 23, 2024
3. Cboe Global Markets: Cboe global markets reports trading volume for december and full year 2023. https://www.nasdaq.com/press-release/cboe-global-markets-reports-trading-volume-for-december-and-full-year-2023-2024-01-04 (2024), accessed: November 23, 2024
4. De Spiegeleer, J., Madan, D.B., Reyners, S., Schoutens, W.: Machine learning for quantitative finance: fast derivative pricing, hedging and fitting. Quantitative Finance **18**(10), 1635–1643 (2018). https://doi.org/10.1080/14697688.2018.1495335, https://doi.org/10.1080/14697688.2018.1495335
5. Flórido, D.P.: Estimate european vanilla option prices using artificial neural networks. https://repositorio.ul.pt/bitstream/10451/53641/1/TM_Diogo_Florido.pdf (2019), accessed: November 23, 2024
6. Funahashi, H.: Artificial neural network for option pricing with and without asymptotic correction. Quantitative Finance **21**(4), 575–592 (2021). https://doi.org/10.1080/14697688.2020.1812702, https://doi.org/10.1080/14697688.2020.1812702
7. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
8. Graves, A.: Supervised Sequence Labelling with Recurrent Neural Networks, Studies in Computational Intelligence, vol. 385 (2012)
9. Hayes, A.: What are greeks in finance and how are they used? https://www.investopedia.com/terms/g/greeks.asp (2023), accessed: November 23, 2024
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)
11. Horvath, B., Muguruza, A., Tomas, M.: Deep learning volatility. IDEAS Working Paper Series from RePEc (2019)
12. Hull, J.C.: Options, Futures, and Other Derivatives. Pearson, 10th edn. (2018)
13. Jarunde, N.: Machine learning for options pricing: Predicting volatility and optimizing strategies – explore how ml models can outperform traditional pricing models (like black-scholes), enhancing option traders' decision-making. Journal of Scientific and Engineering Research **10**(3), JSERBR (2023a)
14. Ke, A., Yang, A.: Option pricing with deep learning. https://cs230.stanford.edu/projects_fall_2019/reports/26260984.pdf, accessed: November 23, 2024
15. Li, W.: Application of machine learning in option pricing: A review. Advances in Economics, Business and Management Research (2022). https://doi.org/10.2991/aebmr.k.220405.035, https://doi.org/10.2991/aebmr.k.220405.035

16. Liu, S., Borovykh, A., Grzelak, L.A., Oosterlee, C.W.: A neural network-based framework for financial model calibration. Journal of Mathematics in Industry **9**(1), 1–28 (2019). https://doi.org/10.1186/s13362-019-0066-7, https://doi.org/10.1186/s13362-019-0066-7
17. Merton, R.C.: Theory of rational option pricing. The Bell Journal of Economics and Management Science **4**(1), 141–183 (1973)
18. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature (1986)
19. Zeroual, A., et al.: Deep learning methods for time series forecasting. Big Data Analytics **25**(3), 152–169 (2020)
20. Zouaoui, H., Naas, M.: Option pricing using deep learning approach based on lstm-gru neural networks: Case of london stock exchange. Data Science in Finance and Economics **3**(3), 267–284 (2023). https://doi.org/10.3934/dsfe.2023016, https://doi.org/10.3934/dsfe.2023016

Appendix

Data Name	Description
underlying_symbol	Ticker symbol of the underlying asset for the option
quote_datetime	Date & time when option quote was recorded
root	Root symbol for the option
expiration	Expiration date of the option contract
strike	Price at which the option can be exercised
option_type	Whether option is "Call" or "Put" type
open	Option's opening price for the trading day
high	Option's highest price for the trading day
low	Option's lowest price for the trading day
close	Last traded price of the option at the end of the trading day
trade_volume	Number of option contracts traded during the trading day
bid_size	Number of contracts available to buy at the bid price
bid	Highest price a buyer is willing to pay for the option
ask_size	Number of contracts available for sale at the ask price
ask	Lowest price a seller is willing to accept for the option
underlying_bid	Highest price a buyer is willing to pay for the underlying asset
underlying_ask	Lowest price a seller is willing to accept for the underlying asset
implied_underlying_price	Calculated price of the underlying asset based on the option market's expectations
active_underlying_price	Current market price of the underlying asset
implied_volatility	Market's forecast of the underlying asset's volatility based on the option price
delta	Derivative of the option price with respect to its underlying price
gamma	Sensitivity of the option price with respect to its delta
theta	Derivative of the option price with respect to time to maturity
vega	Sensitivity of the option price to changes in the implied volatility
rho	Sensitivity of the option price to changes in interest rates
open_interest	Total number of outstanding option contracts not exercised, closed, or expired

Fig. 23: Overview of Dataset Attributes