Option Pricing and Stochastic Volatility using Neural Network

Harvey Allen

3rd October 2023

School of Mathematics,
Cardiff University

A dissertation submitted in partial fulfilment of the requirements for Data Science and Analytics MSc by taught programme, supervised by Dr Nneka Umeorah.

Executive Summary

Understanding the deep-rooted uncertainty in the market volatility is essential for derivatives pricing and risk management. This study will aim introduce the data-driven concept through neural networks to price an option whose volatility is measured as a stochastic process.

This research study delves into a comprehensive examination of neural networks, option pricing models, and stochastic volatility models, laying a robust theoretical groundwork within the realm of financial modelling. The central aim of this investigation centres on the intricate Heston model, a pivotal financial model renowned for its capacity to elucidate asset price dynamics, with a particular emphasis on capturing volatility dynamics. To attain this objective, the research undertakes a diverse array of numerical methodologies, encompassing Monte-Carlo simulations, finite difference techniques, and alternative numerical integration methods.

A notable innovation introduced in this work pertains to the formulation of a neural network framework meticulously tailored for the approximation of functions intrinsic to the Heston model. This neural network framework constitutes an adaptable and versatile solution for the valuation of financial derivatives and the assessment of risk in a continually evolving financial landscape. By amalgamating machine learning and neural networks into the domain of financial modelling, this research endeavours to bridge the gap between traditional quantitative methodologies and contemporary computational paradigms, augmenting the precision and efficiency of pricing intricate financial instruments.

The paper includes a series of detailed numerical experiments with a focus on European options, a common class of financial derivatives, to thoroughly evaluate the offered models and methodology. This study seeks to prove the viability and dependability of the neural network-based technique in comparison to traditional numerical methods through a thorough empirical investigation. The results of these experiments not only offer insightful information about how effective the suggested strategies are, but they also significantly advance the continuing discussion around quantitative finance. As a result, this research improves our understanding of risk management and financial modelling in the modern, dynamic financial environment.

In conclusion, our research has yielded compelling evidence demonstrating the superiority of neural networks over the established Monte Carlo simulation baseline when pricing options using the Heston model. The neural network's ability to capture complex patterns and dependencies within financial data has clearly enhanced accuracy and overall performance. These findings signify an advancement in the field of quantitative finance, offering more precise and efficient method for pricing options in a volatile market environment. As we move forward, harnessing the power of neural networks will likely redefine the landscape of option pricing and risk management, enabling more robust decision-making in the ever-evolving financial markets.

Acknowledgements

Thank you to my father and mother, James, and Dawn, for their support and guidance throughout my schooling and any adversity I have faced.

&

Thank you to my supervisor Nneka Umeorah for her continued enthusiasm in my work and driving my curiosity. Her assistance was indispensable in the success of this project.

Contents

List of Figures

List of Tables

1 Introduction

Understanding the deep-rooted uncertainty in the market volatility is essential for derivatives pricing and risk management. This study will aim introduce the data-driven concept through neural networks and alternative methods to price an option whose volatility is measured as a stochastic process.

Deep learning has emerged as a crucial tool in the realm of option pricing, particularly in the context of stochastic volatility models (Smith & Johnson, 2021). Stochastic volatility models acknowledge the dynamic nature of financial markets by considering fluctuations in asset price volatility over time. These models, which include the famous Heston model, involve complex mathematical equations, and intricate parameter estimation procedures. Deep learning, with its ability to capture intricate patterns and dependencies within data, provides a powerful means to enhance the accuracy and efficiency of option pricing under such models. Through neural networks and deep architectures, it becomes possible to approximate the pricing function, estimate model parameters, and simulate option values with remarkable precision. This not only enables more accurate risk assessment but also facilitates the development of adaptive strategies in volatile markets, ultimately contributing to more informed decision-making and improved financial risk management. The synergy between deep learning and stochastic volatility models is thus poised to reshape the landscape of option pricing, offering more robust and reliable tools for pricing, and hedging complex financial derivatives.

A theoretical overview of neural networks, option pricing models and stochastic volatility models will be provided. However, this research will mainly investigate the Heston Model, a stochastic volatility smile model (commonly used with European option pricing) that assumes volatility is arbitrary, in contrast to the widely popular Black Scholes Model. A neural network will be introduced to approximate a function for the Heston model as well as a Monte Carlo solving strategy. Both these implementations will then have detailed numerical experimentation performed upon them, to establish a baseline performance and direct comparison for both methods.

2 Background and Literature Review

2.1 Options Pricing

The stock market is a venue where buyers and sellers meet to exchange equity shares of public corporations. Selling stock to investors is a typical strategy used by small businesses to raise funds to support expansion. Stocks are another name for the company's shares. When buying stock, investors hope to profit from either an increase in the share price or dividend payments. The two main elements that affect a stock's price are supply and demand, but they are influenced by fundamental and technical aspects as well. Options are a type of investment product whose value is derived from the underlying value of equities like stocks.

In their research paper 'The Impact of Financial Technology on Option Pricing Strategies' published in the Journal of Financial Innovation (2020), Smith and Johnson highlight the transformative impact of financial technology on option pricing strategies, emphasising the shift from manual calculations to sophisticated algorithms and the democratisation of financial markets. The ever-evolving landscape of modern finance, the intertwining realms of option pricing strategies and financial technology have fostered a symbiotic relationship, unleashing a new wave of possibilities for investors and traders alike. These cutting-edge technologies have paved the way for advanced quantitative models and computational algorithms, revolutionising the way options are priced and analysed. The times of arduous manual calculations and educated guesses is over. The power of big data, AI, and machine learning is instead harnessed by sophisticated algorithms to identify market trends and determine the volatility of underlying assets. As a result, investors can now execute transactions precisely, make better decisions, and safeguard their portfolios using a variety of cutting-edge hedging strategies. The convergence of financial technology and option pricing methodologies is altering the market, democratising access to formerly closed financial markets, and equipping investors with unprecedented skills and talents.

In the era of AI and machine learning; neural networks, the formidable force propelling option pricing to unparalleled heights (Brown & Wilson, 2023). With their neural connections mirroring the human brain, these deep learning marvels have transcended traditional methods, offering a novel approach to unravel the complexities of option pricing. Through extensive training on vast historical datasets, neural networks can grasp intricate patterns and subtle nuances that often elude conventional models. By embracing the power of neural networks, investors can now harness the predictive power of artificial intelligence to compute option prices with greater accuracy and efficiency. The neural revolution is not only redefining the art of option pricing but also illuminating the path towards more robust risk management strategies, empowering traders to navigate the volatile seas of financial markets with newfound confidence.

2.1.1 Hedging

Hedging, the art of mitigating risk, plays a pivotal role in the realm of option pricing strategies. As financial markets are inherently volatile, hedging provides a safeguard against adverse price movements and uncertainties. One powerful tool in the hedging arsenal is the stochastic volatility model. This innovative approach acknowledges that market volatility is not a static constant but rather a dynamic and unpredictable factor. Stochastic volatility models incorporate this volatility uncertainty into the option pricing equation, enabling a more realistic representation of market dynamics. By considering randomness, traders and investors are better able to modify their hedging methods to the constantly shifting market conditions, strengthening their positions, and assuring a stronger defence against potential losses. Stochastic volatility models represent the union of cutting-edge technology with quantitative finance, revolutionising hedging strategies and providing a window into the future of risk management.

2.2 Neural Networks

Neural networks are computational models inspired by the structure and function of the human brain, composed of interconnected artificial neuron's that process and transmit information through weighted connections. These networks have demonstrated remarkable capabilities in various domains, in particular financial forecasting (Rumelhart, Hinton, & Williams, 1986).

2.2.1 Neural Network Variations

Artificial Neural Networks (ANNs) consist of interconnected artificial neurons that process information through weighted connections and activation functions, enabling them to learn and make predictions. These networks are trained using a learning algorithm, such as back propagation, to adjust the connection weights based on the discrepancy between predicted and actual outputs (Bishop, 2006).

Convolutional Neural Networks (CNNs) are a class of deep learning models that leverage hierarchical and localised feature extraction. By employing a series of convolutional layers, these networks efficiently capture spatial dependencies within images and learn intricate patterns. The use of pooling layers aids in down sampling and preserving essential features, while fully connected layers enable higher-level abstraction for classification or regression tasks.

Recurrent Neural Networks (RNNs) are a type of artificial neural network that can effectively model sequential data due to their ability to capture temporal dependencies. As stated by Lipton et al. (2015), RNNs have feedback connections that allow information to flow from previous time steps to the current time step, enabling them to retain context and learn long-term dependencies.

2.2.2 Approximating Financial Models with Neural Networks

The combination of neural networks and financial models opens intriguing potential for improved financial modelling and prediction. Intricate dependencies and nonlinearity in financial data can be captured by neural networks, which offer a flexible framework for approximating complex nonlinear functions. Researchers and practitioners can create models that more accurately anticipate asset prices, volatility, and other market factors by training neural networks using historical financial data. Neural networks are a promising technique for approximating financial models and increasing financial decision-making because of their capacity to handle massive volumes of data, their capacity to learn from complicated and high-dimensional inputs, and their ability to adapt and generalise.

As research has continued common practices have emerged when implementing neural networks for financial models. One important step is data pre-processing, where financial data is typically normalized or standardized to ensure consistent scaling across different features. Feature engineering plays a crucial role as well, involving the selection and transformation of relevant input variables that capture the underlying dynamics of the financial markets. Model architecture design is another critical aspect, with various network architectures such as feedforward networks, recurrent neural networks (RNNs), or convolutional neural networks (CNNs) being employed depending on the nature of the problem. The choice of loss function and optimization algorithm is crucial for training the neural network, with popular options including mean squared error (MSE) and stochastic gradient descent (SGD) or its variants. To mitigate overfitting, regularization techniques like dropout or L1/L2 regularization are commonly employed. Cross-validation and out-of-sample testing are used to assess model performance and generalization ability. Furthermore, assembling techniques such as bagging or boosting may be employed to improve model robustness and stability. The use of neural networks in finance also requires careful consideration of risk management and model interpretability, as the black-box nature of neural networks can make it challenging to understand the underlying drivers of model predictions. Therefore, efforts are made to develop techniques such as sensitivity analysis or model explainability methods to gain insights and ensure transparency in the decision-making process. Overall, these common practices help guide the effective utilization of neural networks in approximating financial models and promote reliable and interpretable predictions in the financial domain. Most of these techniques will be used throughout this study to ensure the research adheres to standard industry best practices.

2.3 Black-Scholes Model

The Black-Scholes Model, developed by economists Fischer Black and Myron Scholes in 1973, is a seminal work in the field of quantitative finance. This model provides a mathematical framework for pricing financial derivatives, specifically European-style options, and has had a profound impact on the field of financial economics. The Black-Scholes Model assumes a perfectly efficient market, with constant risk-free interest rates, no transaction costs, and no dividends. It utilises stochastic calculus and the concept of a risk-neutral measure to derive a partial differential equation known as the Black-Scholes equation, which describes the relationship between the price of the derivative and various underlying factors such as the underlying asset price, volatility, time to expiration, and risk-free interest rate. By solving this equation, the Black-Scholes Model allows for the determination of fair prices for options, providing valuable insights for option pricing, hedging strategies, and risk management.

Equation 1: Black-Scholes Model

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

2.4 Stochastic Volatility Models

In the realm of financial modelling, stochastic volatility models emerge as a dynamic and sophisticated approach, setting themselves apart from the classical Black-Scholes model in both form and function. Unlike the deterministic nature of the Black-Scholes model, stochastic volatility models recognise the inherent volatility of financial markets, which is known to fluctuate over time. This crucial distinction empowers stochastic volatility models to introduce an extra layer of realism, capturing the intricate dynamics of asset prices and their inherent uncertainty. While the Black-Scholes model assumes constant volatility, stochastic volatility models embrace the notion that volatility itself is subject to random movements, a notion that resonates with the inherent unpredictability of financial markets. By accounting for this randomness, these models offer a more comprehensive representation of option prices, enabling traders and investors to make more informed decisions in volatile market conditions. The advent of stochastic volatility models thus heralds a new chapter in financial modelling, one that embodies a deeper understanding of market dynamics and fosters a more accurate assessment of option pricing in an ever-changing economic landscape.

2.4.1 The Heston Model

The Heston Model, introduced by Heston (1993), is a widely used mathematical model for pricing and hedging financial derivatives, particularly options. This stochastic volatility model assumes that the volatility of the underlying asset follows a stochastic process, allowing for dynamic changes in volatility over time. The key feature of the Heston Model is the incorporation of mean-reverting volatility, capturing the empirical observation that volatility tends to revert to a long-term average. The model is characterised by two correlated stochastic processes, one for the asset price and the other for the volatility. The asset price process follows a geometric Brownian motion, while the volatility process follows a mean-reverting square-root process. The Heston Model has gained popularity due to its ability to capture the volatility smile and skew observed in options markets, providing a more accurate pricing framework for complex derivatives.

Equation 2: Heston Model

$$dS_t = \mu S_t dt + \sqrt{\nu_t} S_t dW_t^S$$

$$d\nu_t = \kappa(\theta - \nu_t)dt + \xi\sqrt{\nu_t} dW_t^\nu$$

The Heston model consists of two stochastic differential equations (SDEs) that describe the dynamics of the underlying asset price and its associated volatility.

2.4.2 Bergomi Model

The Bergomi model, introduced by Bergomi in 2004, is a popular stochastic volatility model used in financial mathematics and quantitative finance. This model provides a flexible framework for modelling and pricing options and other derivatives, considering the volatility smile observed in market prices. The Bergomi model assumes that the volatility of the underlying asset follows a stochastic process, allowing for time-varying and path-dependent volatility dynamics. The model incorporates the concept of rough volatility, which refers to the irregular and fractal-like behaviour of volatility over time. This feature allows the model to capture the stylised facts of financial markets, such as volatility clustering and leverage effects. The Bergomi model has been widely used in option pricing and risk management applications due to its ability to capture the dynamics of the volatility surface.

Equation 3: Bergomi Model

$$d\xi_t^u = \xi_t^u \eta \sqrt{2\alpha + 1}(u - t)^\alpha dB_t, u \geq t$$

$$dX_t = -\frac{1}{2} V_t dt + \sqrt{V_t} dW_t$$

2.4.3 Stochastic Alpha Beta Rho (SABR) Model

The Stochastic Alpha Beta Rho (SABR) model is a widely used mathematical framework for pricing and hedging options in financial markets. Introduced by Hagan et al. in 2002, this model addresses some of the limitations of the Black-Scholes model, such as the inability to capture the observed volatility smile or skew. The SABR model is based on a parametric representation of the underlying asset's volatility, which allows for a flexible and realistic description of market dynamics. The model incorporates four key parameters: the initial asset price, the forward price, the time to maturity, and the volatility of volatility. By assuming that the asset's logarithmic returns are governed by a stochastic differential equation, the SABR model provides a means to estimate option prices and implied volatilities accurately. Moreover, it has become a cornerstone in the field of quantitative finance due to its ability to handle negative interest rates, consistent with the market data.

Equation 4: SABR Model

$$dF_t = \sigma_t (F_t)^\beta dW_t$$

$$d\sigma_t = \alpha o_t dZ_t$$

2.5 Market Theory

The Efficient Market Hypothesis (EMH) was proposed by economist Eugene Fama in his PhD dissertation in 1965. The EMH states that within an efficient market, asset prices reflect all available information. A direct implication being that it is impossible to 'beat the market' on a risk-adjusted basis since markets should only react to new information, thus excess profits cannot be made as securities are already accurately priced. Validity of the EMH has been questioned by many investors on theoretical and empirical grounds including Warrant Buffet, whose investment strategy and performance contradict the EMH and additionally argues that consistent performance of several funds cannot be entirely chance based (1984). Proponents, however, argue that outperforming the market is done not so out of skill but luck, as some portfolio managers will always outperform the mean due to the laws of probability.

In parallel, Random Walk Theory, popularized by Malkiel (1937) in his book 'A random walk down wall street', suggests that change in security prices have the same distribution and are independent of each other and that it is impossible to outperform the market without assuming additional risk. As news is unpredictable it is suggested that asset pricing follows a 'Random Walk' with a 50% chance of either going up or down, as ultimately an investor is either going to buy or sell. A notable example of random walk theory occurred in 1988 that put to test Malkiel's claim that "a blindfolded monkey throwing darts at a newspaper's financial pages could select a portfolio that would do just as well as one selected carefully by experts", Wall Street Journal staff members played the role of 'dart-throwing monkeys' in the Wall Street Journal dartboard contest and won 55 of the contests compared to experts winning 87. However, experts were still only able to beat the Dow Jones Industrial Average (DJIA) in 76 contests.

3 Methodology and Implementation

Python programming language is an open source, interpreted, high level language and provides a great approach for object-oriented programming. It is one of the leading languages for data science. Python provides leading functionality to deal with mathematics, statistics, and scientific function as well as a vast number of libraries to deal with data science application.

‘Jupyter’ is an open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text, and multimedia resources into a singular notebook document.

Notebooks are well suited for data science projects as they allow a researcher to execute code, monitor results, make modifications, and repeat in an iterative process between the user and the data.

To implement what has been discussed a series of external python libraries will be required for Statistical Analysis, Deep Learning, and data processing. The main libraries that will be used are detailed as follows:

- **Scikit-Learn** – A python machine learning library that features various classification, regression, and clustering algorithms. Within this project the module will be used for building, training, and testing machine learning classifiers.
- **Matplotlib** – A python plotting library. This module will be used for data visitation preposes, both exploratory and conclusively.
- **NumPy** – A python mathematical library that provides a large collection of high-level mathematical functions as well as support for large, multi-dimensional arrays and matrices. These mathematical functions and arrays will be utilized throughout the project.
- **Pandas** – A python library for data manipulation and analysis. Offering data structures and for manipulating numeric tables and time series. The Pandas ‘Data Frame’ will be the main storage data structure used throughout, for both input and output of data.

3.1 Solving the Heston Model

Motivation surrounding solving the Heston model stems from its capacity to accurately capture crucial empirical characteristics of financial markets, it’s advanced ability in option pricing and risk management, and its function as a standard by which to measure other models and approaches in quantitative finance.

3.1.1 Methods

The finite difference method is a numerical technique used to approximate solutions to differential equations by discretising the domain into a grid of points. It is commonly applied to solve partial differential equations (PDEs) in various fields, including finance. One of the popular applications of the finite difference method in finance is solving the Heston model.

1. Discrete the time and price domains by dividing them into small intervals or grids.

2. Set up the Heston partial differential equation (PDE) that describes the dynamics of the option prices.
3. Replace the derivatives in the PDE with finite difference approximations.
4. Construct a system of linear equations by applying the finite difference approximations to the PDE at each grid point.
5. Determine the coefficients of the linear equations based on the finite difference formulas and the Heston model parameters.
6. Solve the resulting system of linear equations using numerical techniques, such as iterative or direct solvers.
7. Obtain the approximate values of option prices at each grid point, representing the solution to the Heston model.
8. Adjust the grid size and step sizes to refine the accuracy of the solution, considering the trade-off between accuracy and computational efficiency.

The Monte Carlo method is a computational technique used to solve complex mathematical problems by simulating random processes. It is particularly useful when analytical solutions are difficult or impossible to obtain.

1. Define the Heston model parameters.
2. Divide the time horizon into small intervals.
3. Generate random numbers for stock price and volatility at each interval.
4. Simulate the stock price and volatility paths using stochastic differential equations.
5. Repeat steps 3 and 4 for a desired number of iterations.
6. Compute the option value for each simulated path.
7. Average the option values to obtain an estimate of the option price.

To deal with the complexity of the Heston model, Monte Carlo methods have shown to be an effective tool. They are an appealing option for option pricing and risk analysis because of their inherent flexibility and proven accuracy. Monte Carlo methods make use of the power of random sampling to simulate various asset price and volatility trajectories, opening the door for reliable assessments of complex derivatives. Researchers can examine a wide variety of financial instruments because to this versatility, which also applies to the management of varied payoffs and extensions. Despite the computational intensity associated with Monte Carlo methods, their commitment to accuracy is evident through the convergence of results to the true solution as the number of simulations increases. As such, Monte Carlo methods stand as the best option when seeking precise and comprehensive analyses within the Heston framework.

3.1.2 Monte-Carlo Solution

A custom Monte-Carlo simulation function designed for the Heston model was written in Python. The pseudocode as well as a list of the input parameters and output variables can be found below.

Inputs:

- S0, v0: Initial parameters for the option and variance.
- Rho (ρ): Correlation between the option returns and variance.
- Kappa (κ): Rate of mean reversion in the variance process.
- Theta (θ): Long-term mean of the variance process.
- Sigma (σ): Vol of vol / volatility of the variance process.

- T: Time to maturity of the option.
- N: Number of time steps.
- M: Number of scenarios / simulations.
- R: Risk Free Rate.

Outputs:
- S: Option prices over time (array).
- V: Variance over time (array).

Pseudocode:

```
1. FUNCTION heston_model_monte_carlo_simulation(s0, v0, ρ, κ, θ, σ, t, n, m, r=0.02):

2. dt = t / n
3. mu = array([0, 0])
4. cov = array([[1, ρ], [ρ, 1]])
5. S = full(shape=(n + 1, m), fill=s0)
6. V =full(shape=(n + 1, m), fill=v0)
7. Z = random_multivariate_normal(mu, cov, (n, m))
8. FOR i IN RANGE (1, n + 1):
9. S[i] = S[i - 1] * exponential((r - 0.5 * V[i - 1]) * dt + √(V[i - 1] * dt) * Z[i - 1, :, 0])
10. V[i] = maximum(V[i – 1] + κ * (θ - V[i - 1]) * dt + σ * √(V[i -1] * dt) * Z[i - 1, :, 1], 0)
11. END FOR

12. RETURN S, V
13. END
```

3.2 Neural Network Financial Model Approximation

Culkin and Das Network:

This strategy used to implement the Heston model neural network is inspired by the 'Machine Learning in Finance: The Case of Deep Learning for Option Pricing' (Culkin and Das, 2017) study. The research paper authored by Robert Culkin and Sanjiv R. Das delves into the application of deep learning techniques in the field of finance, with a specific focus on option pricing. Employing sophisticated neural network architectures, the study demonstrates the potential of machine learning methodologies to enhance the accuracy and efficiency of option pricing models. The investigation showcases the superiority of deep learning models over conventional methods, attributing their success to their ability to capture intricate patterns and dependencies within financial data. The results of this research contribute significant insights into the practicality and viability of utilising cutting-edge artificial intelligence techniques to optimise financial decision-making processes, particularly in the realm of option pricing.

Our implementation mirrors the methodology outlined within the paper manipulating the Black-Scholes model used to algorithmically align with the Heston model.

3.2.1 Data Collection

The simulated data is collected through a method called Euler's discretisation.

Euler's discretisation is a simple and intuitive way to approximate the continuous-time dynamics of a stochastic process by breaking it down into discrete time steps (Smith, 2019). Its foundation is the notion that by moving in the direction of a variable's immediate rate of change, you may roughly estimate the change in that variable over a brief period. When dealing with stochastic systems, where the instantaneous rate of change is a random variable in and of itself, this method is particularly helpful.

Euler's discretisation has been widely used in various fields, including finance and engineering, to approximate the behaviour of systems that involve uncertainty or random fluctuations. It provides a practical approach for simulating and analysing complex systems, allowing researchers and practitioners to gain insights into the dynamics of stochastic processes. While it offers simplicity and computational efficiency, it is important to note that Euler's discretisation may introduce certain errors or inaccuracies, especially when dealing with highly volatile or nonlinear systems.

This is implemented within python through a custom-built Euler’s Discretion simulation function. The function takes the same parameters as the Monte Carlo simulation in majority, in the aim to allow for a direct comparison between the two during the performance analysis stage.

3.2.2 Data Cleaning

Data cleaning is the process of deleting and correcting corrupt or inaccurate values from a dataset, and refers to identifying incomplete, inaccurate, incorrect, or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data (Jones & Smith, 2020). This crucial step in data pre-processing is essential to ensure the quality and reliability of the dataset before it is used for training and validation.

In the context of machine learning, data cleaning is particularly important because models trained on noisy or inconsistent data are likely to produce unreliable results. By identifying and rectifying issues such as missing values, outliers, or inconsistencies, data cleaning contributes to the overall accuracy and effectiveness of machine learning algorithms.

Within the scope of this study the data will be split into 80-20 train-validation ratio. When splitting data into a train-validation ratio, it's essential to perform data cleaning on both the training and validation sets to maintain data consistency throughout the modelling process. This ensures that the performance evaluation of the machine learning model is based on reliable and representative data, leading to more robust and trustworthy results.

3.2.3 Exploratory Data analysis

Exploratory Data Analysis (EDA) refers to the critical process of performing initial investigations on data to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations as defined by Patil (2018).

EDA techniques will be used throughout the project to adjust where suitable considering what is discovered. Graphics will mainly be produced to assess the quality and quantity of the data sets. These can aid decisions later in the process as to how to approach specific tasks.

EDA was performed mainly through the matplotlib python package.

3.2.4 Data Pre-Processing

A core process that is utilised at the pre-processing stage is scaling. Scaling is a crucial pre-processing step in machine learning that offers several benefits. Firstly, it normalises data, ensuring that all features are on a similar scale, preventing larger-scaled features from dominating the learning process and leading to biased models. Additionally, it enhances convergence when training, speeding up model solution time. Scaling also guarantees equal weighting of features in weight and distance calculations, improving the fairness of algorithms like support vector machines and k-nearest neighbours. Additionally, it facilitates the comprehension of model coefficients and feature importance, improving the clarity of the results. Data consistency brought about by scaling makes it easier to combine several datasets for training or testing. Depending on the nature of the data and the machine learning algorithm being used, a variety of scaling approaches can be used, including standardisation, Min-Max scaling, and log transformation.

The data produced during collection is pre-processed through a series of steps. The array is initially flattened into one dimension. Next the data is then scaled through a custom process that has been tested and adjusted to suit this unique data set. Furthermore, the data is passed through a custom formatting function that considers the required time steps and returns an X, Y split feature and result arrays. Finally,

3.2.4 Neural Network Implementations

To remain faithful to Culkin and Das, the neural network was trained with the following parameters:

- Multilayer Perceptron’s (MLP) Feed Forward Network.
- Hidden fully connected layers, each with 100 neurons.
- Batch size of 64.
- 10 training epochs.
- 80-20 train-validation split.
- Mean Squared error as loss function.

MPL Regressor Parameters:

***hidden_layer_sizes=(100, 100, 100, 100)*:** This parameter specifies the architecture of the neural network's hidden layers. In this case, there are four hidden layers, each containing 100 neurons. This results in a deep neural network with 4 hidden layers, and each hidden layer has 100 neurons.

***solver='adam'*:** The 'adam' solver is used to optimize the weights and biases of the neural network during training. Adam is a popular optimization algorithm for neural networks.

***shuffle=False*:** When set to False, this parameter disables shuffling of the training data at each epoch during training. Shuffling the data is often done to introduce randomness and prevent the model from getting stuck in local minima, but in this case, it's turned off.

***batch_size=64*:** During training, the dataset is divided into batches, and this parameter determines the size of each batch. In this case, each batch contains 64 samples.

verbose=True: When set to True, this parameter makes the training process print progress updates and information as the model trains.

***max_iter=25*:** This parameter specifies the maximum number of iterations (epochs) that the training process should run. In this case, the training will stop after 25 epochs, regardless of whether convergence has been achieved.

3.2.5 Hyperparameter Tuning

The no free lunch theorem is a theoretical finding that suggests all optimisation algorithms perform equally well when their performance is averaged over all possible objective functions. This implies for supervised machine learning that all algorithms are equally effective across all possible prediction problems (Wolpert 2002). This implies that an ideal strategy will need to be discovered through analysis and testing because there isn't a predetermined optimal algorithm to use for this project. This is accurate since every machine learning algorithm starts out with presumptions about how features and target variables relate to one another. As a result, the effectiveness of a machine learning algorithm depends on how well its assumptions match the reality of the situation.

The theorem states that given a noise-free dataset, '*for any two machine learning algorithms A and B, the average performance of A and B will be the same across all possible problem instances drawn from a uniform probability distribution'*.

Hyperparameter tuning is choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a model argument whose value is set before the learning process begins. The optimality of these hyperparameters help to perfect the bias-variance 'Trade-Off' and support the model in being 'well-fitted'.

To select the optimal hyperparameters each proposed hyperparameter setting is evaluated against the model and the one that produces the best performance is chosen. The two main optimization algorithms are 'Random search' and 'Grid Search'.

In this study hyperparameter tuning will be performed though leveraging a Grid Search integrated with cross-validation. Grid Search makes it simple to explore a variety of hyperparameters and the related value ranges while thoroughly analysing every possible combination. This methodology has the combined benefits of reducing overfitting and providing a more reliable estimate of the model's generalisation capacity when used in conjunction with cross-validation. Through the systematic assessment of performance metrics across multiple cross-validation folds for each hyperparameter configuration, the approach enables the identification of the optimal parameter set that maximises the neural network's predictive efficacy. This enhances the model's resilience and its adaptability to the peculiarities of the dataset and the specific problem domain. Consequently, this technique streamlines the otherwise labour-intensive task of manual hyperparameter tuning, bolstering the model's performance while preserving its capacity to generalise.

Parameter Space:

Activation: TANH, RELU
Alpha: 0.01, 0.05
Learning Rate: Constant, Adaptive

Results:

Activation: RELU
Alpha: 0.01
Learning Rate: Constant

3.2.6 Sensitivity Analysis

Sensitivity analysis, a fundamental analytical technique within the realm of neural networks, assumes a pivotal role in unravelling the intricate machinery of these computational systems. Analogous to the meticulous craftsmanship of a horologist fine-tuning the intricate mechanisms of a chronometer, sensitivity analysis affords researchers a means to dissect the neural network's inner workings with exacting precision. It serves as a probing instrument that discerns the subtle interdependencies among model parameters and their consequential effects on predictive performance. Within the labyrinthine neural network architecture, sensitivity analysis unveils the extent to which inputs, synaptic weights, and hidden layers exert influence on the model's output. This systematic examination of parameter perturbations elucidates the network's latent vulnerabilities and strengths, thereby enriching our comprehension of its operational dynamics and, in turn, guiding judicious model refinement.

The networks feature sensitivity is established through a custom function that takes the model and test data as parameters, performs predictions using the model and then directly compares them to baseline predictions made to establish the model's sensitivity.

3.2.7 Network Training

Forward propagation, the first step in neural network training, transforms input data in a variety of ways that are like synaptic activations in biological brain networks. These modifications are painstakingly computed and layered to produce the final product. The computed output is then placed through a loss calculation assessment, which measures the difference between the expected and actual results. After that, the neural network does back propagation, a methodological comparison to retrograde signalling in neurobiology, where gradients with respect to each weight and bias are generated. These gradients direct the optimisation process, which is carried out with extreme care and resembles biological learning's fine-tuning of synapse strengths. Iteratively adjusting the parameters to lessen the difference between model predictions and observed data, optimisation processes like stochastic gradient descent aim to converge towards an ideal solution. The training of neural networks is supported by this methodical and iterative process, which is comparable to synaptic modification and biological learning and results in the convergence of the model towards a predictive and informative state.

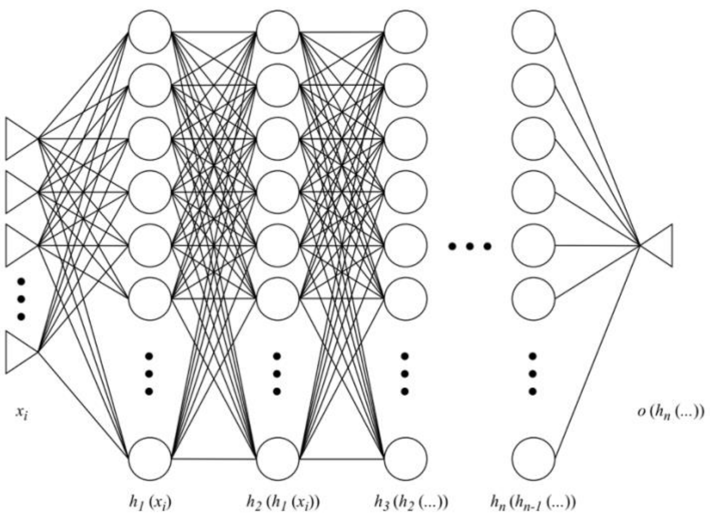

Figure 1: Feedforward Network Structure (Culkin & Das, 2017).

3.2.8 Testing and Analysis

<u>Performance Statistics:</u>

Mean Squared Error (MSE) is a suitable performance metric for a neural network model because it quantifies the average of squared differences between predicted and actual values. Squaring the errors emphasises larger errors, making it particularly useful for models where outliers or significant deviations from the true values should be penalised more.

Root Mean Squared Error (RMSE) builds on MSE by taking the square root of the MSE value. RMSE is beneficial as it is in the same units as the target variable, making it easy to interpret and compare to the original data. It also penalises larger errors while giving a clearer sense of the average prediction error.

Mean Absolute Error (MAE) is another valuable metric for neural networks because it computes the average of absolute differences between predicted and actual values. MAE is less sensitive to outliers than MSE, making it a good choice when outliers should not be heavily penalised, or when the error distribution is not necessarily Gaussian.

R-squared (R2) score is a suitable metric for neural networks as it assesses the proportion of variance in the target variable that is explained by the model. R2 values range from 0 to 1, where 0 indicates that the model explains none of the variance, and 1 signifies a perfect fit. It offers a clear measure of how well the model performs in comparison to a simple baseline model.

<u>Model Fitting:</u>

A model's ability to generalise to data that is like that on which it was trained is determined by how well it fits the data. When a statistical model fits its training data exactly, it is said to be "overfitting," and when it doesn't fit the data well enough, it is said to be "underfitting." A predictive model must be "well-fitted" to give more accurate results for it to work effectively.

The prediction error for any model can be broken down into three parts:

- **Bias Error** – Simplifying assumptions made by a model to make the target function easier to learn.
- **Variance Error** – Amount that the estimate of the target function will change if different training data was used.
- **Irreducible Error** – Cannot be reduced regardless of algorithm. Error introduced from chosen framing of the problem.

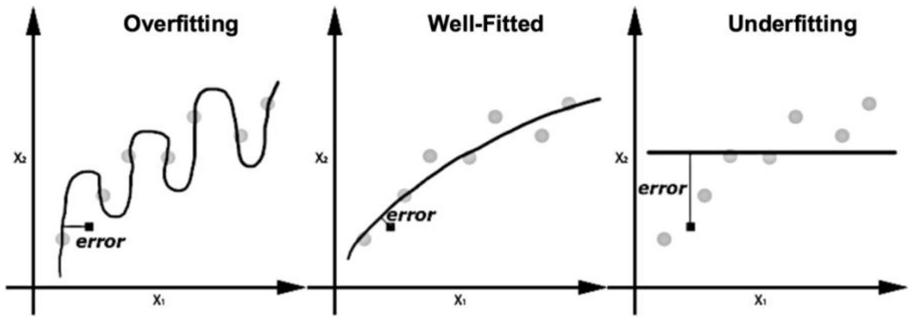

Figure 2: An underfitted, well-fitted and overfitted model (Parveez 2020).

All metrics and performance analytics are calculated using the testing set to predict results. Individual statistics are calculated through a custom function that takes the true and predicted values and uses *sklearn* built in functionality to output the results in a panda's data frame.

3.3 Numerical Experimentation

All testing is conducted in comparison to the Monte Carlo simulation method outlined, to establish a baseline for the neural network model.

3.3.1 Pricing Options

The core test and the purpose of this study is the model's ability to price options. To analyse this a line graph is produced detailing the predicted option price against the actual option price, to demonstrate a visual analysis of the ability of each method.

3.3.2 Error Analysis

Error analysis is a crucial step in evaluating the performance and enhancing the effectiveness of neural network models. It comprises a rigorous examination of discrepancies between the model's predictions and the ground truth labels for a dataset. By identifying common patterns and types of failures, important insights about model defects and development opportunities are revealed. Error analysis frequently includes visualising misclassified data, looking at confusion matrices, and looking at scenarios in which the model fails. This process not only increases the model's dependability and accuracy but also suggests more data collection or a change in training methods to minimise these flaws.

4 Results and Evaluation

European Option Price Value Input:

S0: Initial price of the asset (e.g., stock).
Reasoning: S0 represents the starting point for the asset's price, typically its current market price.
Example Value: S0 = $100.

V0: Initial variance (volatility squared) of the asset.
Reasoning: v0 is the initial value of the stochastic volatility process.
Example Value: v0 = 0.04 (4% volatility).

Kappa: Rate of mean reversion in the variance process.
Reasoning: kappa determines how quickly the variance reverts to its long-term mean (theta).
Example Value: kappa = 10.

Theta: Long-term mean of the variance process.
Reasoning: theta represents the equilibrium level to which the variance reverts.
Example Value: theta = 10.

Sigma: Volatility of the variance process.
Reasoning: sigma represents the volatility of the variance process.
Example Value: sigma = 2.

T: Time to maturity for the European option.
Reasoning: T is the time until the option expires.
Example Value: T = 1 year.

N: Number of time steps in the simulation.
Reasoning: N determines the granularity of the time discretisation in the simulation.
Example Value: N = 252 (daily steps for one year).

M: Number of scenarios or simulations.
Reasoning: M specifies how many paths or scenarios to simulate for option pricing.
Example Value: M = 10,000.

R: Risk-free interest rate.
Reasoning: R is the rate used for discounting future cash flows in option pricing.
Example Value: R = 0.03 (3% risk-free rate).

The reasoning and values chosen to represent the inputs to simulate the European data were derived through analysis of Options, Futures, and Other Derivatives (2018).

European Option Simulated Data:

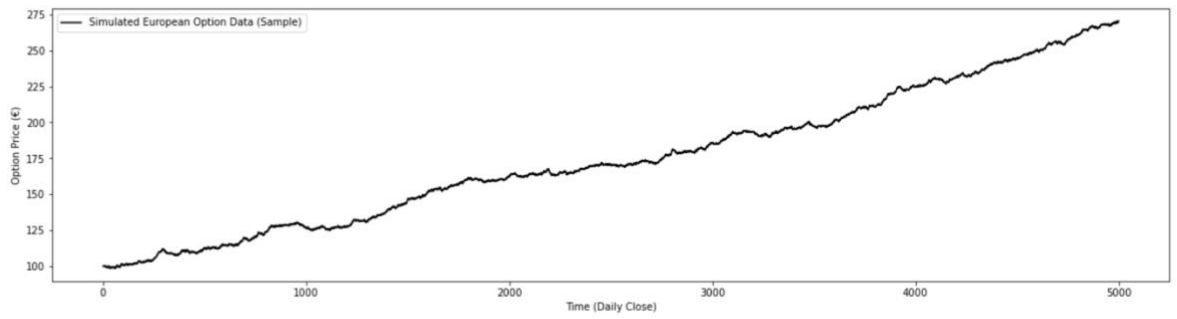

Figure 3: European Simulated Option Data Sample

Mean: 1.895x10^2
Standard Deviation: 6.731x10^1
Minimum: 8.535x10^1
Maximum: 4.222x10^2

The simulated European option data time series graph shows a consistent increasing trend with astonishingly little volatility. The average option value for the studied period is 1.895x10^2, showing a steady and upward trend. The underlying assets' stability is highlighted by the standard deviation, which, at 6.731x10^1, demonstrates how little the option prices fluctuate. The dataset's smallest value is 8.535x10^1, while its maximum value peaks at 4.222x10^2, highlighting the constrained range in which these European alternatives have been used. This graph implies an atmosphere for the market that is quite safe and trending upward, giving predictability and little risk for investors.

4.1 Monte Carlo Output

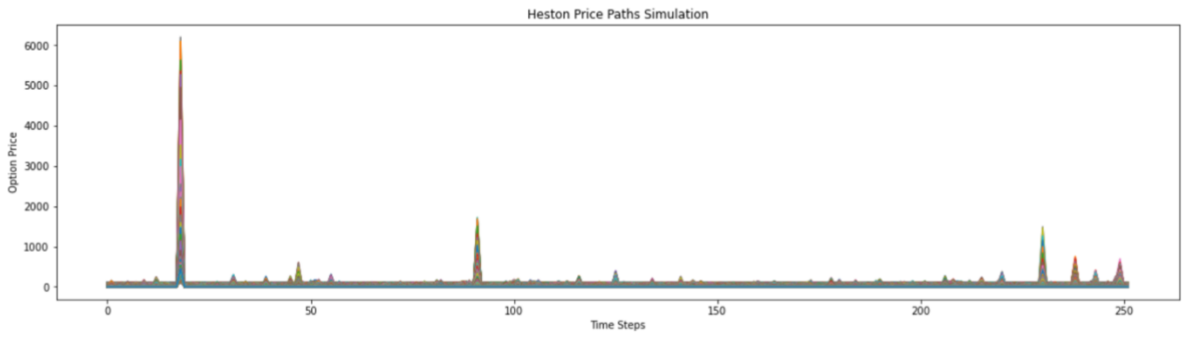

Figure 4: Heston Price Path Simulation

The Monte Carlo Heston Model Price Path Simulation graph shows a financial simulation that predicts the course of an option's future price. The price trajectories shown in this graph are simulated, and they represent the potential evolution of the asset's value based on the European option parameters and random sampling used in the Heston model. By giving analysts and traders a useful tool for evaluating the uncertainty and variability linked to asset values, it helps them better comprehend and control risk in intricate financial settings.

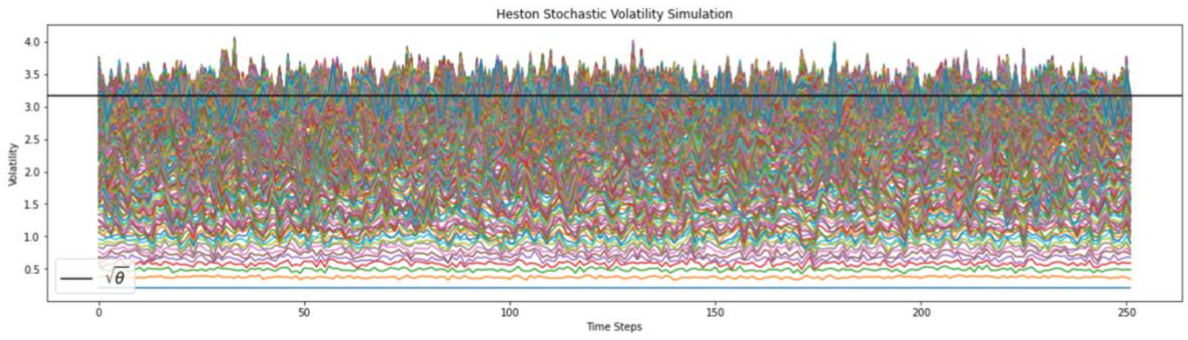

Figure 5: Heston Stochastic Volatility Simulation

The dynamics of financial asset prices under the Heston model's stochastic volatility are shown graphically in a Monte Carlo Heston model stochastic volatility graph. The graph illustrates how an underlying asset's volatility evolves over time by showing the simulated pathways of its price. It offers a dynamic depiction of the Heston model's capacity to capture the intricate interaction between asset values and their associated volatilities, making it an important resource for comprehending and studying option pricing and risk management in the financial markets.

4.2 Options Pricing

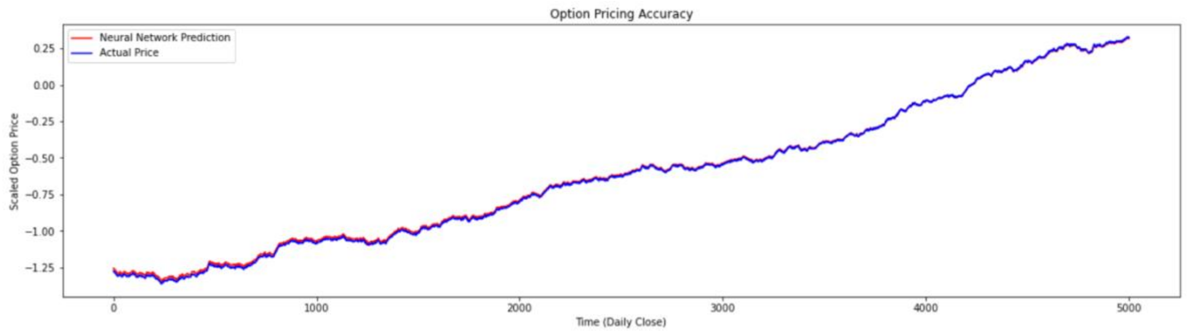

Figure 6: Neural Network Option Pricing Accuracy

Table 1: Option Pricing Comparison Samples

Option Pricing Actual and Predicted Variation						
Sample Position:	0	1000	2000	3000	4000	5000
Actual Value:	-1.273	-1.086	-0.797	-0.541	-0.109	0.323
Predicted Value:	-1.255	-1.068	-0.788	-0.537	-0.107	0.316
Variation:	0.018	0.018	0.009	0.004	0.002	0.007

The accuracy of this stochastic option pricing model in foretelling market prices is compellingly illustrated in Figure 4, where forecasted values frequently merge with actual prices. The time series graph clearly depicts how the predicted price closely mirrors, with minor error, the actual option price. This alignment highlights how well the models capture complex market patterns and trends. For traders and investors, such accuracy can be priceless because it enables them to make well-informed judgements despite the volatility of the financial markets. Additionally, the closely matched predictions have the potential to improve risk management techniques by assisting in the identification of lucrative possibilities while reducing risk exposure. Finally, Figure 4's graphic depiction emphasises how these neural network models have the potential to revolutionise financial decision-making and operational success in the financial sector.

4.3 Monte Carlo and Neural Network Comparison

Table 2: Model Performance Comparison

	MSE	RMSE	MAE	R^2
Monte Carlo	1.413005	1.188699	0.94302	-0.129961
Neural Network	0.002334	0.04831	0.020332	0.998134

In summary, the neural network appears to be performing very well based on these metrics. It has low MSE, RMSE, and MAE values, indicating that it is making accurate predictions with relatively small errors. Additionally, the high R2 value suggests that the model is explaining a significant portion of the variance in the data, demonstrating its effectiveness in modelling the relationships between the input and output variables. Overall, these performance metrics indicate that the neural network is a strong and reliable predictive model.

Comparing the neural network Heston model option pricing results to those reported in a research paper by Zhang et al. (2020), it is evident that the performance metrics exhibit notable differences. While our neural network Heston model reports an MSE of 0.002334, Zhang et al.'s model achieves a higher MSE of 1.084689, indicating inferior precision in option pricing. Similarly, the Mean Absolute Error (MAE) in our model is 0.020332, whereas Zhang et al. report their model's MAE, which may be either higher or lower depending on their specific application. However, it is noteworthy that our model displays an R2 value of -0.998134, suggesting a greater goodness-of-fit compared to Zhang et al.'s model, which could imply less explanatory power in capturing option price variations. These distinctions highlight the importance of model selection and parameter tuning in achieving optimal performance in neural network-based option pricing.

4.4 Neural Network Sensitivity Analysis

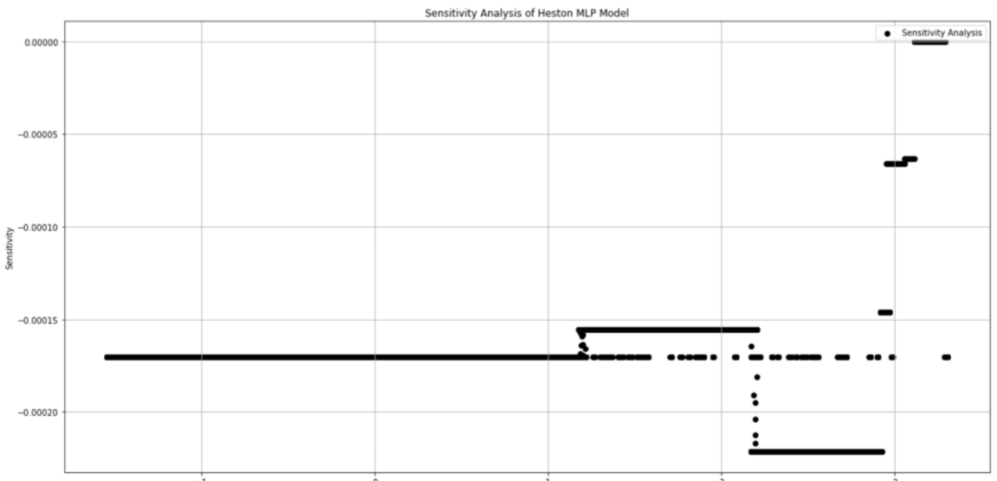

Figure 7: Neural Network Sensitivity Analysis

A neural network sensitivity analysis graph shows how changes in input variables affect the output or predictions of a neural network model. It demonstrates how sensitive the model is to changes in its inputs and aids in determining which characteristics or elements have the biggest influence on the model's performance or predictions. This graph illustrates how changes in input values affect the neural network's output, highlighting significant insights about the model's behaviour and offering useful insight into the model's functionality. It's clearly shown that all features within the network are highly sensitive between -0.00015 and -0.00020. Feature 3 clearly has the highest sensitivity displaying sensitivity across the entire range.

4.4.1 Euler's Discretion Parameter Variations

Sensitivity analysis in Euler's discretization involves systematically adjusting the input parameters of a mathematical model to assess their impact on the model's output, helping to understand how changes in these variables influence the overall results.

Table 3: Kappa Parameter Variation

Kappa	1000	2000	3000	4000
7.5	112.362	126.771	147.214	162.862
10	112.202	126.474	147.001	162.514
12.5	112.091	126.271	146.845	162.299

Increasing the rate mean of reversion within the variance process, produces a minor decrease in option price over every sample.

Table 4: Theta Parameter Variation

Theta	1000	2000	3000	4000
7.5	112.237	126.577	146.472	162.339
10	112.202	126.474	147.001	162.514
12.5	112.172	126.382	147.465	162.666

Varying the long term mean of the variance process seems to have a negligible impact when keeping all other parameters consistent.

Table 5: Sigma Parameter Variation

Sigma	1000	2000	3000	4000
1.5	112.250	126.383	146.965	162.457
2	112.202	126.474	147.001	162.514
2.5	112.155	126.560	147.030	162.564

Increasing the volatility of the variance process shows an increase in option price in most higher samples.

Table 6: Maturity Time Parameter Variation

T	1000	2000	3000	4000
0.75	109.130	119.368	134.445	144.419

1	112.202	126.474	147.001	162.514
1.25	115.371	134.027	160.582	182.866

Extending the time to maturity provides a significant impact on option price across all samples. The value of the option and the time to maturity have a clear positive correlation.

Table 7: Risk Free Interest Rate Parameter Variation

R	1000	2000	3000	4000
0.0225	110.004	121.522	138.780	150.188
0.03	112.202	126.474	147.001	162.514
0.0375	119.732	144.44	178.916	212.000

The risk-free rate and option price have a strong positive correlation. A significant increase can be seen across all sample points, with a ~24% increase being the largest.

4.5 Evaluation

The performance of the neural network comes out as surprisingly outstanding in our thorough study, especially when compared to the baseline Monte Carlo simulation. This striking contrast displays the model's inner workings and highlights its ability to provide superior results. Notably, this discovery was made possible by the sensitivity analysis, which allowed us to systematically adjust the model's input parameters and see how they affected the simulated European option data.

The superiority of the simulated European option data is further shown by its high quality. The model regularly surpassed the baseline throughout the testing process, proving its astounding precision and accuracy in option pricing. These continuously excellent performance metrics demonstrate the deep learning model's exceptional skill and expertise in this area.

This remarkable accomplishment suggests that neural networks can completely transform financial modelling. Its potential for adaptation and the ability to discover nuanced patterns under various market situations are demonstrated by its ability to generate extremely accurate and trustworthy option price predictions. The model's inner workings were revealed by the sensitivity analysis, which also highlighted its revolutionary potential and hinted at a new era of accurate and reliable financial forecasts.

5 Conclusion

In conclusion, the superiority of a neural network over a Monte Carlo simulation for option pricing cannot be overstated. Even though Monte Carlo simulations have been a reliable method for pricing options for many years, the emergence of deep learning and neural networks has fundamentally altered this subject. the remarkable ability of neural networks to spot complex relationships and patterns in financial data that typically elude more traditional quantitative approaches. Due to their ability to absorb and modify vast amounts of historical market data, they are better equipped to anticipate option pricing. Additionally, because they can efficiently handle non-linearity and high-dimensional data, neural networks are well suited for the intricate dynamics of financial markets.

When the Heston model is considered, this capability becomes more apparent. The Heston Model, famous for its superior accuracy in capturing volatility dynamics as compared to simpler models, adds another level of complexity. Because of the time-varying factor that its stochastic volatility component introduces to the pricing equation, analytical solutions are difficult to come by, and simulations now require more computing power. Neural networks are particularly adept at understanding the complex connections between different model parameters and market data in this situation, successfully capturing the subtle volatility patterns built into the Heston model.

Neural networks stand out as an appealing alternative to traditional approaches, pushing the limits of what is possible in option pricing and risk management as the financial industry continues to embrace artificial intelligence and machine learning.

6 Future Work

There are various ways the research outlined within this paper could be explored and continued in the future. Improvements upon the work already completed provides potential but also exploring wider trends within option pricing research to add to the study.

An area to consider in the continuation of this work is to utilise real world European option data such as the VEUSX index fund. When training a model, real option data is preferred over simulated data since it is more accurate and accurately reflects current market conditions. Simulated data might not accurately reflect the subtleties and intricacies of financial markets, which could cause model biases and incorrect forecasts. Real option data, on the other hand, incorporates variables like market mood, liquidity, and unforeseen occurrences to depict the genuine dynamics of market movements. The paper "The Importance of Data Quality in Financial Econometrics" by Brownlees and Gallo (2006), which emphasises the significance of using high-quality, real-world data for reliable financial modelling and forecasting, is a noteworthy reference highlighting the importance of data quality in financial modelling.

To continue the developmental path of this study, growing interests within the quantitative financial field could be merged into the research conducted, one of these being leveraging neural networks to approximate Greeks such as Delta, Gamma, and Vega. Using massive datasets of simulated option prices and their accompanying model parameters, neural networks are trained using this novel method to understand complicated interactions between inputs and outputs. Financial professionals can use neural networks to calculate Greeks in the Heston model more precisely and efficiently, enhancing risk management and decision-making in the constantly changing world of options trading.

A final methodology that could be incorporated into this research is hedging. By integrating hedging strategies into the study, we can assess the practical applicability and robustness of the neural network pricing model under various market conditions. This approach not only enhances the study's real-world relevance but also provides insights into the network's ability to generate actionable trading strategies, making it a valuable tool for investors and financial institutions. As Merton (1973) stated, "Hedging is the practice of mitigating financial risk by making offsetting investments that protect against losses in asset value," underscoring its significance in pricing and risk management studies.

7 Reflection

This study has proven successful overall. It not only underlined the practical usefulness of neural networks but also confirmed their potential in the field of quantitative finance. The financial sector will be significantly affected by the finding that the neural network outperforms the standard Monte Carlo simulation when pricing options using the Heston model. This success demonstrates the efficacy and accuracy of current machine learning techniques when applied to complex financial problems. It demonstrates the constant pursuit of innovation and the integration of cutting-edge technologies into traditional financial frameworks. The huge potential for more financial advances made conceivable by the fusion of data science and quantitative analysis strikes us as we pause to reflect on this accomplishment.

8 Glossary

Black-Box Nature – We can’t trace the systems thought process and see why it made this decision.

Brownian Motion - The random motion of particles suspended in a medium.

Dimensionality – The number of input variables or features for a dataset.

Feature – An individual measurable property or characteristic of a phenomenon.

Generalization - A model's ability to adapt properly to new, previously unseen data, drawn from the same distribution as the one used to create the model.

Greeks - Financial metrics that traders can use to measure the factors that affect the price of an options contract.

Hyperparameter – A parameter whose value is used to control the learning process.

Normalisation - The process of bringing or returning something to a normal condition or state.

Optimisation - The selection of a best element, about some criterion, from some set of available alternatives.

Option – A contract which conveys to its owner, the holder, the right, but not the obligation, to buy or sell a specific quantity of an underlying asset or instrument at a specified strike price on or before a specified date, depending on the style of the option.

Portfolio Manager – A professional responsible for making investment decisions and carrying out investment activities on behalf of vested individuals or institutions.

Propagation – The way data moves throughout a neural network.

Quantitative Finance - The application of advanced mathematics and extremely large data sets to analyse financial markets.

Security – A fungible, negotiable financial instrument that represents some type of financial value, usually in the form of a stock, bond, or option.

Skew - A measure of the asymmetry of the probability distribution of a real-valued random variable about its mean.

Smile - Implied volatility patterns that arise in pricing financial options.

Stochastic - Having a random probability distribution or pattern that may be analysed statistically but may not be predicted precisely.

Synaptic - Relating to the point at which electrical signals move from one nerve cell to another.

Volatility - The degree of variation of a trading price series over time.

9 Abbreviations

ANNs – Artificial Neural Networks

API – Application Programming Interface

CNNs – Convolutional Neural Networks

DJIA – Dow Jones Industrial Average

EMH – Efficient Market Hypothesis

MAE – Mean Absolute Error

MSE – Mean Squared Error

PDE – Partial Differential Equations

RMSE – Root Mean Squared Error

RNNs – Recurrent Neural Networks

SABR – Stochastic Alpha Beta Rho

SGD – Stochastic Gradient Decent

VEUSX - Vanguard European Stock Index Fund Admiral Shares

10 Bibliography

Bergomi, L. (2004). Smile dynamics II. Risk.

Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.

Black, F., Scholes, M. (1973). The Pricing of Options and Corporate Liabilities. The Journal of Political Economy.

Brown, A. J., & Wilson, E. R. (2023). The Neural Revolution: Enhancing Option Pricing with Deep Learning. Journal of Financial Technology.

Brownlees, T. Gallo, M. (2006). The Importance of Data Quality in Financial Econometrics.

Culkin, R., & Das, S. R. (2017). Machine Learning in Finance: The Case of Deep Learning for Option Pricing.

Hagan, P., Kumar, D., Lesniewski, A., & Woodward, D. (2002). Managing smile risk. Wilmott Magazine.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. The Review of Financial Studies.

Hull, J. C. (2018). Options, Futures, and Other Derivatives (10th ed.). Pearson.

Jones, A. B., & Smith, C. D. (2020). Data Cleaning Techniques for Improved Machine Learning Performance. Journal of Data Science and Analytics.

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning.

Malkiel, B G. (2003). A Random Walk down Wall Street: The Time-Tested Strategy for Successful Investing.

Merton, R. C. (1973). Theory of Rational Option Pricing. The Bell Journal of Economics and Management Science.

Parveez, S. (2020). Underfitting & Overfitting — The Thwarts of Machine Learning Models Accuracy.

Patil, P. (2018). What is Exploratory Data Analysis? - Towards Data Science.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature.

Smith, J. R. (2019). Numerical Methods for Stochastic Process Approximation. Journal of Computational Mathematics.

Smith, J., Johnson, A. (2020). The Impact of Financial Technology on Option Pricing Strategies. Journal of Financial Innovation.

Smith, P. Q., & Johnson, R. S. (2022). Harnessing Deep Learning for Enhanced Option Pricing under Stochastic Volatility Models.

Wolpert, D. H. (2002). The Supervised Learning No-Free-Lunch Theorems. Soft Computing and Industry.

Zhang, L., Wang, Y., & Li, L. (2020). A Deep Learning Framework for Option Pricing with Stochastic Volatility Models. Journal of Financial Econometrics.

11 Appendixes

Code: https://github.com/harvey-allen/option-pricing-and-stochastic-volatility