ERASMUS UNIVERSITY ROTTERDAM

Erasmus School of Economics

Master Thesis Econometrics [Quantitative Finance]

Implied Volatility Surface Prediction Using Non-parametric Approach

Author:
Michael Dihardjo
572734

Supervisor:
Dr. G. Freire
Second Assessor:
Prof. Dr. DJC van Dijk

September 14, 2022

The content of this thesis is the sole responsibility of the author and does not reflect the view of the supervisor, second assessor, Erasmus School of Economics or Erasmus University.

Contents

Abstract

Implied volatility is widely used by participants in the financial market, such as in derivatives pricing by market makers, risk management by portfolio managers, or even individual investors in daily trading. Hence, an accurate prediction of the implied volatility surface could be instrumental for participants in the financial market. This study evaluates non-parametric models in implied volatility surface prediction. There will be two different approaches to modeling the IVS. First, using long short-term memory neural (LSTM) and LightGBM, we portray the problem as a time series prediction problem. In the second approach, using feedforward artificial neural network (ANN) and LightGBM, implied volatility is modeled directly as a function of moneyness, time-to-maturity, and other features without taking into account temporal dependencies. Results indicate that LightGBM outperforms LSTM in the first approach. Similarly, LightGBM outperforms ANN in the second approach.

1 Introduction

Volatility is a measure of dispersion, or in technical terms, the squared deviation from the mean. Volatility is an important measure in the financial industry and is especially useful for investors, as it serves as one of the indicators of risk, which greatly supports the process of painting market conditions. In financial literature, we can typically differentiate between realized volatility and implied volatility. Realized volatility is calculated according to its technical definitions using asset return data, while implied volatility is derived from its derivatives (typically options). Both measures are widely used by participants in the financial market, such as in derivatives pricing by market makers, risk management by portfolio managers, or even individual investors in daily trading. Hence, an accurate prediction of the implied volatility surface could be instrumental for participants in the financial market. In its seminal paper, Black and Scholes (1973) proposed an option pricing model that provides a closed-form relationship between the option price, price of the underlying, strike price, time-to-maturity, interest rate, and volatility. The implied volatility surface (IVS) is a mapping of an asset's implied volatility for different strike prices and time-to-maturity. The Black-Scholes model predicts flat implied volatility across different strike prices and time-to-maturity. However, empirically, option prices in the market deviate from the Black-Scholes model's constant implied volatility. Along with the strike price dimension, the misspecification is most often characterized by the smile shape in the implied volatility. Nonetheless, mapping implied volatility using Black-Scholes provides a convenient way to compare options with different strike prices and maturities.

A large body of literature studies implied volatility, particularly in option pricing. Jackwerth and Rubinstein (1996) non-parametrically derive the underlying asset risk-neutral distributions of the S&P 500 European options. Other examples of a non-parametric approach include Ait-Sahalia and Lo (1998) and Wright (2018). Derman and Kani (1994), Dupire et al. (1994), Rubinstein (1994), and Dumas et al. (1998) model the option price as a deterministic function of the moneyness and time-to-maturity. Both Dumas et al. (1998) and Goncalves and Guidolin (2006) model the IVS as a quadratic function of strike price and time to maturity; they also found that parameters are highly unstable, suggesting a time-varying dynamics in the IVS. Additionally, a principal component analysis approach to studying the dynamics of the implied volatility surfaces has also been done (Skiadopoulos et al. (2000); Cont and da Fonseca (2002)). A larger body of literature studies parametric modeling of option prices. Heston (1993) models option prices based on stochastic

volatility in contrast to the constant volatility assumed in the Black and Scholes (1973) model. Similarly, Hagan et al. (2002) proposed a closed-form approximation for the volatility smile with its SABR (Stochastic, Alpha, Beta, Rho) model; the SABR models forward rate and volatility as diffusion processes. More recently, Almeida et al. (2022) attempts to non-parametrically correct the pricing error of popular parametric models, such as the Heston model. Almeida et al. (2022) found that neural networks can correct the pricing error of different parametric models to a similar level, which points out the potential of other non-parametric approaches.

In regards to time-varying dynamics, Corsi (2008) incorporates realized volatility with the different horizons in its Heterogeneous Autoregressive model of Realized Volatility (HAR-RV) to capture strong persistence and long memory properties in realized volatility. The motivation behind Corsi (2008) is heterogeneity in the market that arises from different time horizons. Participants such as market makers trade with a shorter time horizon, while institutional investors are more concerned with trades with a longer horizon. Realized volatility is a measure obtained by summing the squares of log returns and can be easily obtained from market data. Corsi (2008) constructs realized volatility of different horizons in its HAR-RV model to predict realized volatility.

Similar to Almeida et al. (2022), this study investigates non-parametric approaches to modeling the implied volatility surface, specifically machine learning models. There has been limited study on the application of machine learning techniques to implied volatility surface prediction. However, the literature regarding realized volatility is richer. Luong and Dokuchaev (2018) recently applied Random Forest in Realized Volatility forecasting. Similarly, Kim and Won (2018) integrates long short-term memory neural network with GARCH-type models in its forecasting model. The universal approximation theorem states that neural networks can approximate any continuous function. Thus, we consider the feedforward artificial neural network (ANN) and long short-term memory recurrent neural networks. We also consider a type of gradient-boosted decision tree algorithm, LightGBM, which is popular among practitioners in the machine learning community. Additionally, we incorporate realized volatilities of different horizons, as in Corsi (2008) in an attempt to capture the time-varying dynamics.

Daily options data is obtained from the OptionMetrics dataset for the period between February 2, 2016 to April 30, 2019. We focus on SPX (S&P 500 Index) European options traded at the Chicago Board Options Exchange (CBOE). We also utilize high-frequency intra-day data from the New York Stock Exchange (NYSE) Trade and Quote (TAQ) to construct the realized volatility measures. Two approaches are considered to non-parametrically fit the Implied Volatility Surface

(IVS). In the first approach, the IVS is framed as a time series for some moneyness and time-to-maturity; three forecasting horizons are considered: 1-day, 5-day, and 21-day. From the result of this study, LSTM and LightGBM can achieve lower implied volatility root-mean-squared error (IVRMSE) in 21-day ahead prediction compared to Naive (implied volatility on day t) and Black-Scholes (average implied volatility on day t). In 1-day and 5-day ahead predictions, both LSTM and LightGBM achieve lower IVRMSE compared to Black-Scholes, but not against Naive prediction. Additionally, there is an observable improvement in LSTM when a longer timestep is implemented. In the second approach, we fit the feedforward neural network (ANN) and LightGBM to map implied volatility from a set of features (including moneyness and time-to-maturity). The models are fitted to all available options each day for a given period, and then we see how well they generalize to the future. On average, LightGBM outperforms ANN in terms of out-of-sample IVRMSE. From the result of this study, we see the potential of neural networks and gradient-boosted decision trees in option pricing.

2 Methodology

2.1 Black-Scholes

Black and Scholes (1973) introduced its closed-form relation between option price (C or P), underlying price (S_t), strike price (K), time-to-maturity (τ), volatility (σ) and interest rate(r). The model assumes that the asset price S_t follows a stochastic process,

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{1}$$

where μ is the long-term mean of the asset, σ is the mean and instantaneous volatility of the asset, and Wt is a Wiener process, which essentially is a continuous random walk. Using $Itô's$ calculus, the Black-Scholes partial differential equation for a call option C can be expressed as follows,

$$rS_t\frac{\partial C}{\partial S} + \frac{\partial C}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} - rC = 0 \tag{2}$$

The solution for a call option price is given by,

$$C(S,t) = S_t\Phi(d_1) - Ke^{-r\tau}\Phi(d_2) \tag{3}$$

$$d_1 = \frac{log(\frac{S_t}{K}) + (r + \sigma^2/2)(\tau)}{\sigma\sqrt{\tau}}$$

$$d_2 = d_1 - \sigma\sqrt{\tau},$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. From the market, we can observe option prices and their underlying price, assuming the risk-free rate r is known, the only unobserved variable is σ. Hence, for a call option with strike price K, time-to-maturity τ, we can solve for σ according to Equation 3. This is the implied volatility, and it provides a convenient way to compare options prices with different strike prices and time-to-maturity.

2.2 Realized Volatility

Realized Volatility is the realized quadratic variation of the asset price, computed based on the historical price of the assets. In an attempt to capture the time-varying dynamics in the implied volatility surface, Realized Volatility measures will be included in the feature space. The HAR-RV model by Corsi (2008) attempts to capture long memory dynamics by considering Realized Volatility over different time horizons. In this study, the returns of an asset will be calculated according to Weighted Average Price (WAP) defined in Equation 4, with data obtained from the NYSE TAQ dataset. First, for each day t, we filter for data points in trading hours, between 9:30 AM to 4:00 PM. Next, we calculate WAP for every timestamp with a non-zero $BidSize$ and $AskSize$. The logarithmic returns are based on the difference of WAP at different times in the data.

$$WAP = \frac{BidPrice * AskSize + AskPrice * BidSize}{BidSize + AskSize} \tag{4}$$

Realized Volatility of a given asset can be easily computed from equally spaced returns of the asset. The NYSE TAQ dataset provides very high-frequency data up to milliseconds, giving flexibility in asset return construction. To avoid capturing significant microstructure noise, we construct 5-minute logarithmic returns during trading hours resulting in 78 equally spaced returns $r_{i,t}$, $i = 1, 2, ..., 78$. Realized Volatility with different horizons can then be constructed as the square root of the sum of squares logarithmic returns. In this study, the primary focus will be the end-of-day option panel with t denoting the day of observation. For example, the $60 - minute$ realized volatility on day t is the square root of the sum of squares of the last 12 5-minute logarithmic returns on day t. Since there are 78 5-minute equally spaced returns in each day, 390-minute realized volatility is equivalent to 1-day realized volatility. Realized volatility with periods longer than 1-day includes returns from prior days. Formally, the k-day Realized Volatility on day t, $RV_t^{(k-day)}$ is

$$RV_t^{(k-day)} = \sqrt{\sum_{j=0}^{k-1} \sum_{i=0}^{78} r_{i,t-j}} \tag{5}$$

In this study, in an attempt to capture temporal dynamics we consider realized volatility with horizons: 60-minute, 4-hour, 1-day, 5-day, and 21-day ($RV^{60-minute}, RV^{4-hour}, RV^{1-day}, RV^{5-day}$, RV^{21-day}).

2.3 Model

2.3.1 Feedforward Neural Network (ANN)

Neural networks, also known as Artificial Neural Network (ANN). Neural networks consist of an input layer, hidden layer(s), and an output layer. The hidden layers allow for non-linear transformation, thus allowing a non-linear relationship of mapping from the input to output. Each node in a hidden layer takes a linear combination of outputs from the previous layer and applies an activation function $z(\cdot)$ that produces an output for the next layer. Mathematically, the final output can be written as $g(\sum_j \beta^{(l)} z(\sum_s z...(z(\sum_i \beta^{(1)} x_i)...))$, where g is the output activation function and x_i is the i-th element of input vector X. In theory, the activation function can be any nonlinear function. The final output function $g(\cdot)$ depends on the problem. As we are trying to predict volatility, the output function with output bounds of $[0, \infty)$ is a good candidate. ReLU (Rectified Linear Unit) is essentially a linear function, where negative values are mapped to zero[1]. However, from preliminary testing linear activation actually performs better compared to ReLu. Hence, in this paper $g(\cdot)$ is specified as the linear activation function, while $z(\cdot)$ is specified as ReLU function.

2.3.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory is a neural network encompassed by what is also called recurrent neural network (RNN), popularized by Hochreiter and Schmidhuber (1997). In a feed-forward neural network, signals only travel in one direction from the input to the output layer, whereas recurrent connection in RNN allows cycles between neurons. These recurrent connections allow storing information of prior inputs in order to construct the sequence's next output. In principle, this allows the mapping of prior inputs to outputs, which can be conceptualized as "memory." However, RNNs often suffer from the vanishing gradient (or exploding gradient) problem, where products of subsequent derivatives caused the gradient to become increasingly small, preventing useful weights update. LSTM was designed to tackle this specific problem, where it utilizes the concept of "gates" to better regulate the flow of signals. The input gate regulates input to a layer, while an output

[1]ReLU function is formally expressed as, $R(x) = max(0, x)$

gate regulates the output. Additionally, there is also a forget gate that regulates the information that should be retained. There are variations in the structure of LSTM, however, in this research, the LSTM can be described by the following functions,

$$i_t = \sigma(U_i x_t + V_i h_{t-1} + W_i c_{t-1} + b_i) \tag{6}$$

$$f_t = \sigma(U_f x_t + V_f h_{t-1} + W_f c_{t-1} + b_f) \tag{7}$$

$$c_t = f_t c_{t-1} + i_t \phi(U_c x_t + V_c h_{t-1} + b_c) \tag{8}$$

$$o_t = \sigma(U_o x_t + V_o h_{t-1} + W_o c_t + b_o) \tag{9}$$

$$h_t = o_t \phi(c_t) \tag{10}$$

Here, i_t, f_t, c_t, o_t, and h_t respectively represent the input gate, forget gate, cell state, output gate, and network output (hidden state). U, V, and W with different subscripts represent the weights vectors; b with different subscripts represents the error vectors. In each gate, $\sigma(\cdot)$ represents the sigmoid function, $\sigma(x) = \frac{e^x}{e^x+1}$, which has an output between 0 and 1. The choice follows by looking at the sigmoid function as a filter for signals in each gate. Finally, the latter represents the hyperbolic tangent function, $\phi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$, which has an output between -1 and 1. The $\phi(\cdot)$ function act as the activation function, but other functions can be considered. Figure 1 illustrates an LSTM cell concept.

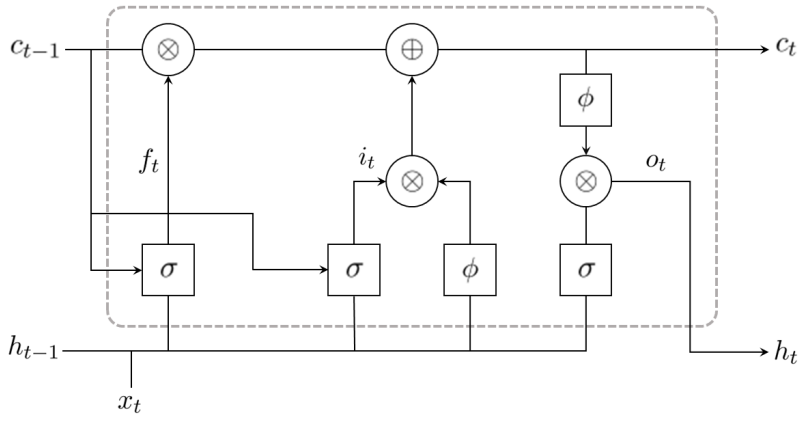

Figure 1: LSTM memory cell illustration

As mentioned, the recurrent structure of LSTM allows cycles of signals between neurons, and the gates are responsible for it. Before going into the gates, we clarify the difference between cell

state c_t and hidden state h_t. The hidden h_t, also present in classic RNN, primarily focuses on signal regarding output at step t. In contrast, the cell state, unique to LSTM, considers signals from the global sequence. The forget gate f_t regulates which information is prioritized and ignored from a cell. Specifically, new input x_t and the hidden state h_{t-1} from prior cell is passed to a sigmoid function. Similarly, the input gate i_t and output gate o_t takes x_t and h_{t-1} before passing it to a sigmoid function, but with different in weights and bias. Additionally, c_{t-1} are also included in f_t, i_t, and o_t based on figure 1. This additionally allows each gate to observe the signal from prior cell state c_{t-1}. Although the gates look similar, they are applied to different signals in sequence. Equation (3) on Figure 1 highlights the different role of f_t and i_t. A gate is simply a vector of numbers ranging between 0 and 1, here, f_t is applied to c_{t-1} deciding signals to keep from the prior cell state, while i_t is applied to the new candidate cell state. Finally, the output gate o_t further decides signals passing through from the resulting cell state c_t for the resulting output h_t.

In application, there are several hyperparameters to be considered in the LSTM model. In this paragraph, we will briefly discuss hyperparameters that are more related to the structure of the network; details of hyperparameters implemented can be found in Section 2.4.1. For those familiar with neural networks, hidden layers and neurons are probably the first hyperparameters that come to mind. Hidden layers are just a layer between input and output, while neurons can be seen as the component that takes inputs, applied weights, and produce output through an activation function. Neural networks have gained success in a wide range of application, however, it is also prone to overfitting. Dropout layers could help avoid overfitting by randomly dropping neurons. Srivastava et al. (2014) points out that adding dropout layers "prevents units from co-adapting too much." In neural networks, weights are adjusted based on the derivative and the effect on the loss function, hence, weights on certain neurons may be updated to compensate for the loss given the weights on other neurons. This form of dependence might lead to overfitting. By randomly dropping the neurons, dependence on the dropped neurons is eliminated. On activation functions, a hyperbolic tangent can be considered, such as ReLU, but in general, we want a bounded function with a derivative that is easy to calculate. Finally, there are other hyperparameters such as, but not limited to learning rate, decay rate, epoch, and batch. However, we will not focus on those in the implementation.

2.3.3 LightGBM

LightGBM, popularized by Ke et al. (2017), is a popular gradient boosted decision tree method. Decision trees resemble a tree-like structure in their decision rules to predict target value. In principle, decision trees partition the feature space into distinct and non-overlapping regions R_j, $j = 1, 2, ..., J$. These regions are constructed by recursively splitting observations based on their features. Each region R_j is then assigned an output value, such that for an input feature $x \in R_j$, the tree prediction is $\hat{y}_j$, which typically is just the average of y_j. Mathematically, a tree can be expressed as

$$T(x; \Theta) = \sum_{j=1}^{J} \hat{y}_j I(x \in R_j), \tag{11}$$

with parameters $\Theta = \{R_j, \hat{y}_j\}_1^J$, and the goal is to minimize,

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{j=1}^{J} \sum_{x_i \in R_j} L(y_i, \hat{y}_j), \tag{12}$$

for some loss function L.

Boosted decision tree methods use an ensemble of decision trees to improve accuracy and robustness. In boosting algorithm, a "weak" learner (decision tree) is sequentially applied to modified data in each step. In the context of a classification problem, at each step, different weights are applied to samples, where larger weights are assigned to misclassified samples. Formally, the boosted tree can be expressed as

$$f_M(x) = \sum_{m=1}^{M} T(x; \Theta) \tag{13}$$

In gradient boosting, we and focuses on sample with large gradient $\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}$ to improve the next decision tree. Borrowing from Hastie et al. (2017), gradient boosted decision tree algorithm can be presented as in Algorithm 1.

Algorithm 1: Gradient Boosted Decision Tree

1. Initialize $f_0(x) = \text{argmin}_{\hat{y}} \sum_{i=1}^{N} L(y_i, \hat{y})$

2. For $m = 1$ to M:

(a) For $i = 1, 2, ..., N$ compute

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$$

(b) Fit a regression tree to the targets r_{im} giving terminal regions R_{jm}, $j = 1, 2, ..., J_m$

(c) For $j = 1, 2, ..., J_m$ compute

$$\hat{y}_{jm} = \underset{\hat{y}}{\text{argmin}} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \hat{y})$$

(d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \hat{y}_{jm} I(x \in R_{jm})$

3. Output $\hat{f}(x) = f_M(x)$

LightGBM employed Gradient-based One-Side Sampling (GOSS) in its sampling algorithm. GOSS focuses on samples with large gradients, the top-a $\times 100\%$ with the largest gradients are selected for subset A, while the remaining samples with small gradients are selected randomly for subset B. Another novel technique in LightGBM is termed Exclusive Feature Bundling (EFB), which essentially reduces the number of features by bundling. For example, in sparse high-dimensional data, mutually exclusive features can be bundled to form a new feature. The concept can also be applied to continuous non-sparse data. EFB in LightGBM does not bundle the raw input features directly, rather, it discretized the data into bins before bundling. Both GOSS and EFB contribute to LightGBM being an efficient and lightweight algorithm, compared to similar gradient boosting decision tree algorithms.

2.4 Implementation

Implementation of the nonparametric models will be divided into two different exercises. In the first exercise, we frame the problem as daily time series where we are interested in the evolution of the implied volatility surface. Daily time series of included as features are $RV^{60-minute}$, RV^{4-hour}, RV^{1-day}, RV^{5-day}, RV^{21-day}, VIX, LTV, LTP, ADS, TMS, and CRS. Using LSTM and LightGBM, we attempt to predict h-day ahead implied volatility for some fixed moneyness and time-to-maturity.

As we are modeling a time series, it is not possible to predict the implied volatility of options with a different strike price and maturity to options to which the model is fitted. However, predicting options with fixed moneyness and time-to-maturity could still give valuable insights into the shape of the implied volatility surface.

For the second exercise, we implemented ANN and LightGBM to learn the implied volatility surface in general. Using historical data, we fit our models to learn the relation between implied volatility and predictor variables: moneyness, time-to-maturity, $RV^{60-minute}$, RV^{4-hour}, RV^{1-day}, RV^{5-day}, RV^{21-day}, VIX, LTV, LTP, ADS, TMS, and CRS. The main difference in the second exercise is that we are not trying to capture any temporal dynamics (apart from the inclusion of Realized Volatility(s) as features), rather we try to model implied volatility in different situations in the market. For both exercises, we evaluate the model based on its out-of-sample implied volatility root mean squared error (IVRMSE).

2.4.1 Forecasting $h - day$ ahead

In the first exercise, we fit the model with the goal to predict the $h-day$ ahead implied volatility(s) for $h = 1, 5, 21$, framing it as a time series problem. Implied volatility data obtained in Section 3.2 formed an imbalanced panel, with each day consisting of a different number of observations with options having different combinations of moneyness and time-to-maturity. As LSTM requires a balanced panel for its input, we will construct a balanced panel of daily implied volatility from daily end-of-day option price data. We consider moneyness in the set $\{0.95, 0.975, 1.0, 1.1, 1, 2, 1.3\}$ and time-to-maturity in the set $\{30, 60, 180, 340\}$ resulting in 24 combinations. Since it is not guaranteed each option is traded for each day, we interpolate the Implied Volatility surface intro grids with discrete points in the moneyness and time-to-maturity dimensions. This exercise of interpolation can be seen as a precursor to generating a 3-dimensional representation of the Implied Volatility surface. We do this by applying a linear interpolation implementation of Watson (2013). Grids for the interpolation consist of 361 discrete points between 0.9 and 1.5 for moneyness and 359 discrete points between 7 and 365 for day-to-expiration. As interpolation heavily depends on starting data points, it is possible that some points in the grid cannot be interpolated. Out of 815 days, we drop 1 day of the time series as it is missing some of the 24 different combinations of moneyness and time-to-maturity. The final result is time series of 24-element vector, denoted as the vector v_t hereinafter, with 814 observations. Plotting the time series illustrates the Implied Volatility skew of the options as shown in Figure 2.

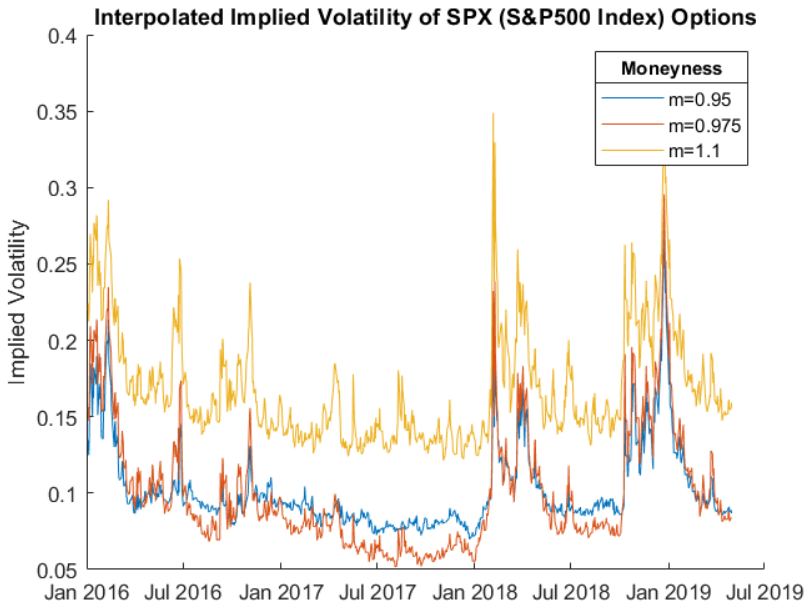

Figure 2: Daily interpolated Implied Volatility of SPX (S&P500 Index) options with time-to-maturity 30 days.

As we are essentially working with time series, we attempt to model the time series evolution. In LSTM, this is implemented by setting the timesteps, indicating the number of past observations used to predict the future. Keep in mind, that increasing the number of timesteps is not equivalent to increasing the number of predictor variables. The timesteps control the length of a single input to output pass. In this study, we consider three different timesteps $= \{3, 21, 42\}$ and three different forecast horizon $h = \{1, 5, 21\}$. For different combinations of timestep and h, different LSTM models will be trained. As mentioned in previous sections, other predictor variables include: $RV^{60-minute}$, RV^{4-hour}, RV^{1-day}, RV^{5-day}, RV^{21-day}, VIX, LTV, LTP, ADS, TMS, and CRS. Tuning neural network hyperparameters is always tricky, as there is an infinite number of combinations of choice. We focus on the choice of hidden layers and neurons, with five different combinations shown in Table 1. Other hyperparameters are set to be the same for all eight LSTM models considered. Each hidden layer is followed by a dropout layer with a 0.2 dropout ratio. In each hidden layer, the recurrent activation function is set to the sigmoid function. Lastly, the final activation function is set to the hyperbolic tangent function. Optimization is done via gradient descent algorithm, popular optimization method includes Stochastic Gradient Descent (SGD) and *Adam* by Kingma and Ba (2014). We chose to set the optimization method to *Adam* for all the models. Prior to fitting

LSTM models, all data is scaled according to min-max scaling, formally expressed in Equation 14.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{14}$$

Table 1: Summary of the eight LSTM models

	LSTM1	LSTM2	LSTM3	LSTM4	LSTM5
Hidden Layers	2	3	4	3	5
Neurons	8, 4	16, 8, 4	32, 16, 8, 4	8, 8, 8	64, 32, 16, 8, 4

This table reports the number of hidden layers and number of neurons in each hidden layer for the five LSTM models. Each hidden layer is followed by a dropout layer with a dropout ratio of 0.2 not shown in the table.

Next, we discuss LightGBM implementation in the first exercise. To model temporal dependence in LightGBM, lagged variables of all features are included as predictor variables. Similar to timesteps in LSTM, we consider up to 3, 21, and 42 lagged observation. Formally, the predictor variables in LightGBM include the Implied Volatility(s) v_t, $RV^{60-minute}$, RV^{4-hour}, RV^{1-day}, RV^{5-day}, RV^{21-day}, VIX, LTV, LTP, ADS, TMS, CRS and their respective lagged value at $t, ..., t-(p-1)$ with $p = \{3, 21, 42\}$. Data scaling is not required in LightGBM as decision trees essentially partition the feature space into different regions, it is not sensitive to monotonic transformation. To model v_{t+h} with LightGBM, we fit LightGBM regressor to the training data to predict each element in v_{t+h} separately, as it does not support multiple output regression. Although for each element in v_{t+h} the LightGBM is fitted separately, we evaluate the model as a whole across all v_{t+h}, that is hyperparameters are the same for the 24 fitted series. We focus on tuning the number of leaves, minimum samples in a leaf, number of boosted trees, maximum tree depth, and learning rate. Leaves here refer to the non-overlapping regions partitioned by the algorithm; perhaps the most important hyperparameter to tune as a sub-optimal choice could lead to overfitting. We apply grid search to find the optimal hyperparameters resulting in the lowest IVRMSE in predicting v_{t+h}. Implementation of grid search in LightGBM is done with the *Optuna* library in python which provides a convenient way to tune hyperparameters. Table 2 summarizes the hyperparameter tuning space in this application. If *step* is not indicated, it means the grid search uniformly samples from the range.

Table 2: LightGBM Hyperparameter Tuning Space First Exercise

	Type	Range	Step
Number of Leaves	Integer	16 to 3184	32
Min. Samples in Leaf	Integer	2 to 500	
Number of Boosted Trees	Integer	100 to 2000	100
Max. Depth	Integer	3 to 12	
Learning Rate	float	0.005 to 0.12	

In regards to cross-validation, since we are working with data where sequence matter, it is not possible to randomly sample during model training. Hence, we will assess the model performance by splitting the data into four different windows. Accounting for the largest $h = 21$ and timesteps $p = 42$, the Implied Volatility series with 814 observations reduces to 752 observations with dates ranging from April 4, 2016 to April 1, 2019 to work with. Four windows of equal length are divided as follows, $window1_{ex1}$ ranging from day 1 to 611, $window2_{ex1}$ ranging from day 48 to 658, $window3_{ex1}$ ranging from day 95 to 705, $window4_{ex1}$ ranging from day 142 to 752. For each window, training and test set are set as the first 564 days and the last 47 days respectively. Finally, the performance of the models is evaluated according to out-of-sample IVRMSE. In addition, two benchmark predictions will be included in the comparison, namely Black-Scholes and Naive prediction. The Black-Scholes model assumes flat volatility, thus Black-Scholes prediction is the average Implied Volatility in the previous day v_t. Naive prediction is defined as the Implied Volatility in the previous day v_t, basically taking the current value as a prediction. The inclusion of Naive prediction is motivated by the high sample autocorrelation property often present in financial data. Figure 3 shows the sample autocorrelation function of an element in the interpolated Implied Volatility series.

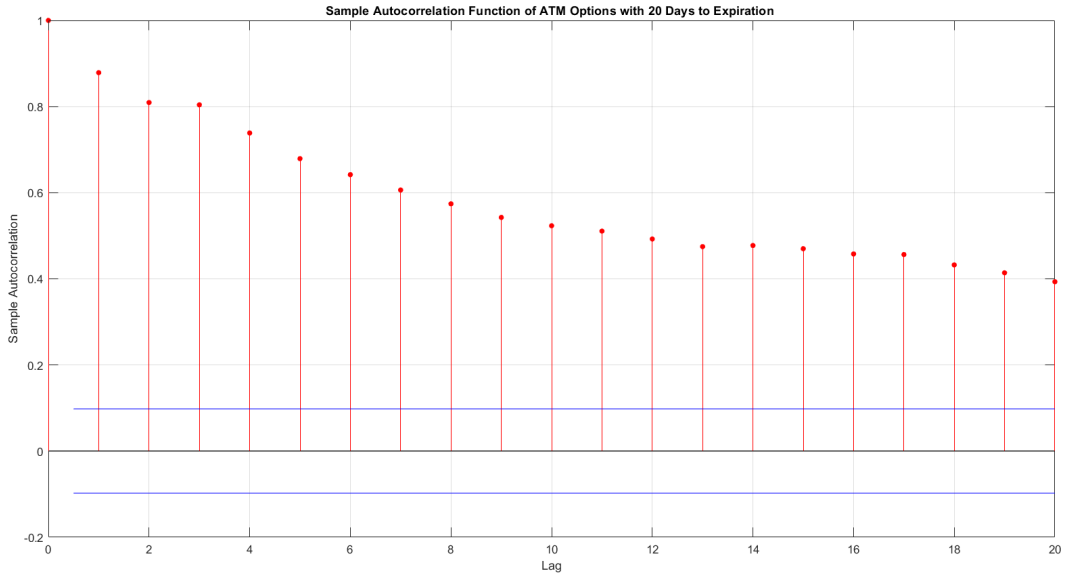

Figure 3: Sample Autocorrelation Function of SPX (S&P500 Index) option

The two benchmarks serve as a purpose to evaluate whether LSTM and LightGBM could make meaningful predictions, rather than just comparing the two.

2.4.2 General Prediction

In the second exercise, we revisit daily Implied Volatility data described in Section 3.2. In principle, we train our models to learn the shape of the Implied Volatility Surface for a period of time and see how well it generalizes in the future. The main difference with the first exercise is we do not model the dependence with past observation directly, rather it is only introduced through the Realized Volatility features.

LSTM model is not included as it required a balanced panel of data. Instead, we include a feedforward neural network (ANN) to compare with LightGBM. Similar to LSTM, min-max scaling will be applied to the data prior to model fitting. We consider five ANN models with different setup summarized in Table 3.

LightGBM hyperparameter tuning is slightly different in the second exercise as the number of observations is much larger; changes can be seen in Table 4. Similar to the first exercise, we split the data into four windows with equal lengths. The first 20 days of daily options data is omitted as RV^{21-day} cannot be calculated. From 815 days, four windows of equal length are divided as follows, $window1_{ex2}$ ranging from day 1 to 611, $window2_{ex2}$ ranging from day 69 to 679,

Table 3: Summary of ANN models

	ANN1	ANN2	ANN3	ANN4	ANN5
Hidden Layers	1	2	3	4	5
Neurons	8	16,8	32,16,8	64,32,16,8	128,64,32,16,8

Table 4: LightGBM Hyperparamter Tuning Space Second Exercise

	Type	Range	Step
Number of Leaves	Integer	16 to 3184	32
Min. Samples in Leaf	Integer	50 to 1000	
Number of Boosted Trees	Integer	100 to 2000	100
Max. Depth	Integer	3 to 12	
Learning Rate	float	0.005 to 0.12	

$window3_{ex2}$ ranging from day 137 to 747, $window4_{ex2}$ ranging from day 205 to 815. The model is trained using all available options available in the first 542 days of each window. Afterward, Implied Volatility predictions are made for all options in the last 68 days of each window. In other words, we train the model to predict an option's implied volatility for given moneyness, time-to-maturity, and features observed. Features included in this exercise are VIX, ADS, LTV, LTP, TMS, CRS, $RV^{60-minute}$, RV^{4-hour}, RV^{1-day}, RV^{5-day}, RV^{21-day}. Again, we include Black-Scholes as a benchmark comparison. Naive prediction is no longer included, as it is not straightforward in this setup, requiring Implied Volatility Surface interpolation with finer grids.

3 Data

3.1 Options and Features

In this section, we discuss the reasoning and source of data being used in the research. More details on implementation will be discussed in the section 2. We primarily focus on options for the underlying S&P500. As of this writing, S&P 500 options are the most actively traded options by volume in the U.S. market, making them a good candidate for studying the implied volatility surface. Numerous studies of option pricing in the past focused on the S&P500 options such as

Goncalves and Guidolin (2006) and Skiadopoulos et al. (2000). Options on the S&P 500 can be traded in the form of SPY options and SPX options. SPY options are derivatives for the S&P 500 ETF (Exchange-traded funds), while SPX options are derivatives for the S&P 500 index. Another key difference is that the former are American-style options, while the latter are European-style options. For the purpose of this research, we will consider SPX options traded at the Chicago Board Options Exchange (CBOE). Daily data of the options with dates ranging from January 4, 2016 to April 30, 2019, constituting 835 trading days, are obtained from the OptionMetrics dataset accessed through Wharton Research Data Services (WRDS). In total, there are 8,698,633 observations before filtering. After filtering, there are 1,655,780 observations, which will be discussed in section 2. As a proxy for the risk-free rate, we will use the 3-month Treasury Bill rate from the Federal Reserve H15 Reports.

As mentioned, we will also utilize the realized volatility of the S&P 500 index as features to possibly capture time-varying dynamics in the IVS. For this purpose, intraday transaction data for the underlying is obtained from the New York Stock Exchange (NYSE) Trade and Quote (TAQ) datasets. The NYSE TAQ datasets provide order book data at very high frequency, up to millisecond accuracy. From the dataset, we construct asset returns from level 1 market data; for more details, go to Section 2. In addition to the constructed realized volatility measures, the VIX index from Chicago Board Options Exchange (CBOE) datasets will also be included. In this study, we use the open VIX index of the day. The VIX index attempts to capture the market's expectation of volatility for the S&P 500 for the next 30 days. The NYSE TAQ, Federal Reserve Bank Reports, and VIX CBOE datasets will all be obtained through WRDS. Similar to Almeida et al. (2022), we also include other features that might be relevant to the problem. Torben Andersen and Viktor Todorov's website, https://www.tailindex.com/, provides Left Tail Volatility (LTV) and Left Tail Probability (LTP). LTV "estimates the expected (risk-neutral) return volatility that stems from large negative price jumps," while LTP "estimates the probability that the index drops by 10% or more over the coming week." LTV and LTP are included with the idea to capture market jump risk, as opposed to the VIX which is more closely related to the volatility diffusion process.

Several features related to macroeconomic conditions are included. The Economic Policy Uncertainty Index (EPU) of Baker et al. (2016) from https://www.policyuncertainty.com quantifies economic condition uncertainty by extracting information from newspapers with different publishers. The Aruoba et al. (2009) business condition index (ADS) is constructed with the idea of capturing current business conditions. ADS index will be obtained from the Federal Reserve Bank

of Philadelphia database. Additionally, the first differences in the term spread (TMS) and the first differences in the credit spread (CRS) are obtained from the St. Louis Federal Reserve Economic Data (FRED) database. TMS is the 10-Year Treasury Constant Maturity Minus 3-Month Treasury Constant Maturity (T10Y3M), Historically, a negative TMS has been viewed as a sign of recession by practitioners. CRS is Moody's Seasoned Baa Corporate Bond Yield Relative to Yield on 10-Year Treasury Constant Maturity (BAA10Y). Corporate bond rates reflect the probability of default. In corporate bonds, AAA-rated bonds are the highest-rated bonds, while BAA-rated bonds are slightly below. Both are seen as investment-grade; however, Baa-rated bonds offer higher returns, which implies higher risk. Hence, the CRS can also be seen as a measure to capture the business condition, since we can see the 10-year Treasury yield as a risk-free rate.

3.2 Implied Volatility

Options data from the OptionMetrics dataset already includes implied volatility; however, we will instead retrieve implied volatility by inverting the Black-Scholes formula. Several initial filters will be applied to the OptionMetrics dataset before further processing. On each day, option prices will be constructed as the midpoint of the end-of-day best bid and end-of-day best ask. Observations with zero volume, zero bid price, zero ask price, and an ask price lower than the bid price are discarded. Next, we filter out options with prices lower than 1/8. For standard options, time-to-maturity τ is calculated as the number of days between the timestamp of the data and the expiration date for each contract, while for AM-settled options, it is the number of days between the timestamp of the data and the expiration date for each contract minus 1. The stock price S_t at the end of each day is the WAP, constructed from the latest order book observation no later than 4 PM on each day from the NYSE TAQ dataset. After obtaining S_t, moneyness can be defined as $m_{i,t} \equiv \frac{S_t}{K}$. Options data are then filtered with respect to time-to-maturity and moneyness; we keep options with a time-to-maturity of at least 6 days and moneyness between 0.8 and 1.6. Furthermore, we only consider out-of-the-money (OTM) options and disregard in-the-money (ITM) options. OTM options are usually priced lower since they have no intrinsic value, making them considerably more liquid compared to ITM options. Additionally, there should be minimal loss of information because of the put-call parity, as Almeida et al. (2022) points out.

As the S&P 500 index typically pays dividends, we estimate q using the put-call parity expressed as,

$$C + Ke^{-r\tau} = P + S_t e^{-q\tau} \tag{15}$$

Solving for q yields, $q = -1/\tau(ln((C - P + KE^{-r\tau})/S_t)$. Interest rate r is the 3-month Treasury Bill rate from Federal Reserve H15 Reports. For each day we compute q for every different time-to-maturity τ using a pair of put and call options that are closest to the money ($m_{i,t}$ closest to 1). We can see the different q as investors' expectations of the asset for the given time horizon τ. Afterward, we further filter the remaining data according to the no-arbitrage condition given below,

$$\begin{aligned} C &\geq max(0,\ S_t - K,\ S_t e^{-q\tau} - Ke^{-r\tau}) \\ P &\geq max(0,\ K - S_t,\ Ke^{-r\tau} - S_t e^{-q\tau}) \end{aligned} \tag{16}$$

Implied volatility is then inverted from the option prices according to Black-Scholes, as in 3. The *RQuantlib* package in R is used to invert the implied volatility for European options.

As briefly mentioned, there will be two different approaches to modeling the IVS. In the first, we portray the problem as a time series prediction problem. In the second approach, implied volatility is modeled directly as a function of moneyness, time-to-maturity, and other features without taking into account temporal dependencies. To have an indication of model performance in different parts of the implied volatility surface, we divide the options based on their time-to-maturity and moneyness. We divide options into three groups of time-to-maturity: SHORT term if $\tau \in [6, 30)$, MEDIUM term if $\tau \in [30, 180)$, and LONG term if $\tau \in [180, \infty)$. In terms of moneyness, we adopt Almeida et al. (2022), so that options are divided into five intervals: deep OTM call (DOTMC) with moneyness if $m_{i,t} \in [0.80, 0.90)$, OTM call (OTMC) if $m_{i,t} \in [0.9, 0.97)$, ATM if $m_{i,t} \in [0.97, 1.03)$, OTM (put) if $m_{i,t} \in [1.03, 1.10)$, and deep OTM put (DOTMP) if $m_{i,t} \in [1.10, 1.60]$.

Table 5 reports a summary of 1,655,780 observations from filtered options data. The Number column indicates the number of observations. Among the most actively traded are MEDIUM-term DOTMP and SHORT-term ATM options with 261,114 and 255,920 observations respectively. The Mean IV column indicates the average implied volatility for each category of options. For all grouping of time-to-maturity, we observe the volatility skew is present and most pronounced in SHORT-term options. According to the Black-Scholes model, higher implied volatility would translate to higher option prices. One possible explanation contributing to the skew could be hedging premiums, especially for DOTMC and DOTMP options. Finally, the Std. dev column indicates the standard deviation of the implied volatility in each category. In general, we see a higher variation in implied volatility as moneyness increases, which indicates time-varying skew of

the implied volatility surface.

Table 5: Summary statistics of S&P 500 Implied Volatility data

	Number			Mean IV			Std. dev		
	SHORT	MEDIUM	LONG	SHORT	MEDIUM	LONG	SHORT	MEDIUM	LONG
Moneyness									
[0.8,0.9)	2840	12466	15646	0.213	0.145	0.120	0.045	0.028	0.021
[0.9,0.97)	83631	134868	26226	0.127	0.107	0.122	0.044	0.031	0.027
[0.97,1.03)	255920	198176	26110	0.116	0.120	0.148	0.047	0.038	0.026
[1.03,1.1)	212581	184287	23871	0.195	0.171	0.174	0.052	0.036	0.024
[1.1,1.6]	147413	261144	70601	0.341	0.276	0.239	0.110	0.070	0.042
All m	702385	790941	162454	0.189	0.182	0.184	0.108	0.086	0.061

This table reports summary statistics of S&P 500 index options Implied Volatility data with dates ranging from January 4, 2016 to April 30, 2019. The columns Number, Mean IV, and Std. dev refers to the number of options, mean implied volatility, and standard deviation of implied volatility. SHORT, MEDIUM, and LONG are the time-to-maturity grouping.

4 Results

4.1 $h - day$ Prediction

Table 6 reports the average IVRMSE across all windows in the first exercise. The column Average reports the average IVRMSE prediction across all h and timestep. LightGBM performs the best on average with Naive prediction performing slightly behind, while Black-Scholes prediction performs significantly worse than the other models. Looking at 1-day ahead prediction, we see that Naive prediction achieves an average IVRMSE of 0.0084 across all windows, outperforming LSTM and LightGBM. Intuitively, this can be explained by the high autocorrelation depicted in Figure 3, that is, we would not expect a large change overnight. Increasing timestep does not result in a noticeable improvement in 1-day ahead forecasting for both LSTM and LightGBM, indicating overfitting of the models. Moving to 5-day ahead, Naive prediction still outperforms any of the non-parametric models. Similarly, both LSTM and LightGBM do not indicate any improvement in 5-day ahead forecasting when the timestep is increased. In contrast, our results indicate improvement in LSTM and LightGBM 21-day ahead forecasting when timestep is increased, presenting

evidence of temporal dynamics captured in a longer sequence of data. On 1-day and 5-day ahead prediction, LightGBM outperforms all LSTM models, while in 21-day ahead prediction, we start to see some LSTM models outperforming LightGBM in terms of IVRMSE. Additionally, both LSTM and LightGBM manage to achieve lower IVRMSE in 21-day ahead prediction. In regards to LSTM, additional hidden layers beyond 4 do not seem to improve LSTM in general, with LSTM5 performing worst on average among all the models.

Performance difference for each model with different timestep can be more easily seen from Table 7, reporting percentage change in IVRMSE when timestep is increased from 3 to 21 or 42. In general, implementing longer timestep= 21 and timestep= 42 does not improve LSTM and LightGBM performance in 1-day and 5-day ahead prediction. Most notably, in 1-day ahead prediction, all non-parametric models considered performed worst as timestep is increased. In 5-day prediction, we also observe implementing a timestep of 21 and 42 does not lead to improvement compared to a timestep of 3. Implementing a longer timestep seems to have the largest positive effect on 21-day ahead prediction for both LSTM and LightGBM, with timestep= 42 generally leading to better performance compared to a smaller timestep. This is evident by looking at both Table 6 and Table 7, with LSTM4 timestep= 42 achieving lowest IVRMSE in 21-day ahead prediction. A possible explanation for this result is that 21-day ahead prediction could be more dependent on a longer sequence of the time series, whereas prediction with a shorter horizon is less dependent on long-term temporal dynamics. Nevertheless, overfitting is also likely in 1-day and 5-day ahead prediction. Overall, our results show that LSTM models require a longer sequence of data to outperform LightGBM. While LSTM models show larger improvement with a longer timestep, LightGBM achieves a lower IVRMSE in general. LSTM outperforms LightGBM in 21-day ahead prediction but requires a more diligent tuning evident from higher variation of IVRMSE in LSTM models shown in Table 6.

Investigating further, we look at the IVRMSE of LightGBM and LSTM1 in the four different windows shown in Table 8. Both LightGBM and LSTM2 have similar patterns in their IVRMSE between different windows, having the largest IVRMSE in the 3rd window. With the current setup, LightGBM and LSTM appear to be sensitive to regime switching in the data. As an illustration of the Implied Volatility skew in the different windows, see Figure 4. This result highlights the problem of time-varying dynamics. Including more observations and features could be beneficial for both LSTM and LightGBM, however, this would exacerbate the already time-consuming training time.

Table 6: IVRMSE Summary in First Exercise

	h=1			h=5			h=21			Average
Timestep	3	21	42	3	21	42	3	21	42	
LSTM1	0.0142	0.0141	0.0142	0.0254	0.0285	0.0284	0.0284	**0.0229**	0.0196	0.0218
LSTM2	0.0156	0.0151	0.0180	0.0250	0.0359	0.0272	0.0333	0.0273	0.0212	0.0243
LSTM3	0.0144	0.0176	0.0207	0.0238	0.0261	0.0308	0.0278	0.0245	0.0291	0.0239
LSTM4	0.0142	0.0149	0.0188	0.0264	0.0309	0.0284	0.0311	0.0232	**0.0188**	0.0229
LSTM5	0.0142	0.0192	0.0227	0.0253	0.0317	0.0269	0.0316	0.0292	0.0256	0.0252
LightGBM	0.0127	0.0139	0.0144	0.0218	0.0240	0.0243	**0.0245**	0.0256	0.0242	0.0206
Naïve	**0.0084**	**0.0084**	**0.0084**	**0.0172**	**0.0172**	**0.0172**	0.0275	0.0275	0.0275	0.0177
Black-Scholes	0.0745	0.0745	0.0745	0.0757	0.0757	0.0757	0.0787	0.0787	0.0787	0.0763

This table reports the IVRMSE summary of model predictions considered in the first exercise. Timestep denotes the different timesteps implemented in the models. The columns $h = 1$, $h = 5$, $h = 21$, denotes h-day prediction IVRMSE. The column Average denotes the average IVRMSE across all h fitted with different timesteps.

Table 7: IVRMSE percentage with different timestep implemented

	h=1			h=5			h=21		
Timestep	3	21	42	3	21	42	3	21	42
LSTM1	-	-1.4%	0.0%	-	12.0%	11.7%	-	-19.2%	-30.9%
LSTM2	-	-3.0%	15.6%	-	44.0%	9.0%	-	-18.0%	-36.3%
LSTM3	-	22.3%	43.7%	-	9.6%	29.2%	-	-12.1%	4.5%
LSTM4	-	4.9%	32.6%	-	16.9%	7.4%	-	-25.5%	-39.6%
LSTM5	-	35.4%	59.9%	-	25.2%	6.2%	-	-7.8%	-19.2%
LightGBM	-	9.5%	13.4%	-	10.2%	11.6%	-	4.5%	-1.2%

This table reports the IVRMSE percentage change of each model when timestep is increased. A positive value means an increase in IVRMSE, while a negative value indicates a decrease in IVRMSE.

Table 8: Comparison of LightGBM and LSTM1 IVRMSE in different windows for the first exercise

	LightGBM				LSTM1			
Window	1st	2nd	3rd	4th	1st	2nd	3rd	4th
Timesteps=3								
h=1	0.0055	0.0122	0.0238	0.0092	0.0075	0.0134	0.0259	0.0102
h=5	0.0049	0.0137	0.0289	0.0080	0.0117	0.0311	0.0413	0.0177
h=21	0.0054	0.0125	0.0272	0.0123	0.0310	0.0273	0.0351	0.0201
Timesteps=21								
h=1	0.0095	0.0235	0.0377	0.0165	0.0078	0.0143	0.0255	0.0086
h=5	0.0088	0.0227	0.0426	0.0220	0.0212	0.0324	0.0360	0.0243
h=21	0.0117	0.0191	0.0447	0.0217	0.0279	0.0263	0.0279	0.0097
Timesteps=42								
h=1	0.0077	0.0379	0.0330	0.0194	0.0081	0.0147	0.0266	0.0076
h=5	0.0105	0.0411	0.0375	0.0133	0.0218	0.0178	0.0432	0.0308
h=21	0.0082	0.0451	0.0316	0.0119	0.0131	0.0209	0.0308	0.0137

This table reports the IVRMSE of LightGBM and LSTM1 in four different windows for the first exercise.

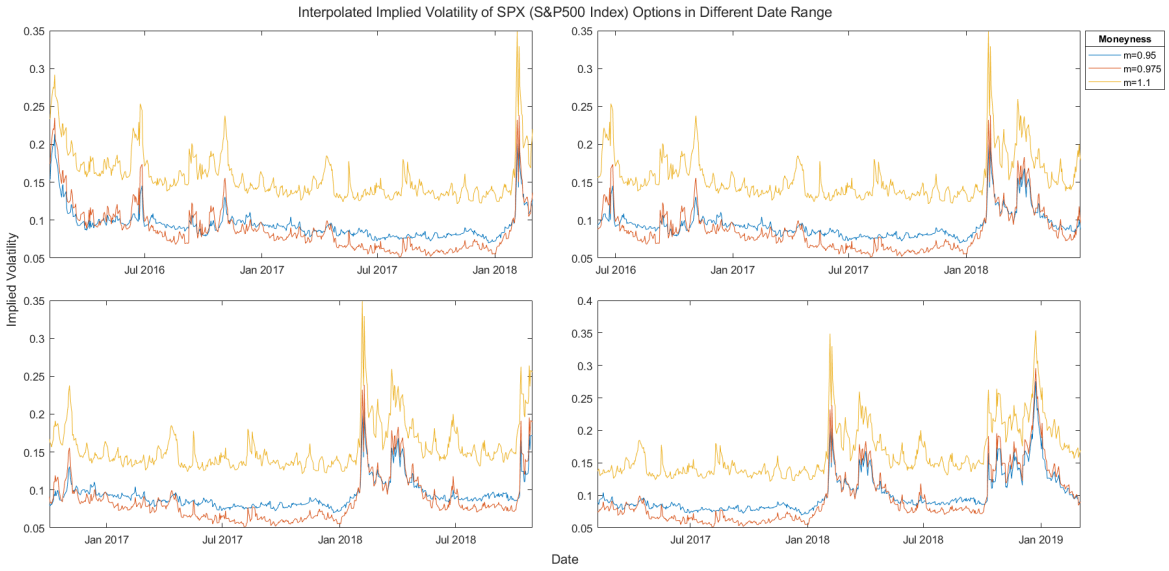

Figure 4: Daily interpolated Implied Volatility of SPX (S&P 500 Index) options with time-to-maturity 30 days in four different windows.

4.2 General Prediction

Table 9 shows IVRMSE summary of each models considered in the second exercise. All ANN and LightGBM models achieved lower average IVRMSE compared to Black-Scholes flat volatility prediction. In exception to ANN4, we see a detrimental effect for each addition of a hidden layer in ANN, failing to improve the model's ability to capture additional non-linear relations in the IVS. However, focusing on DOTMC options, we see large improvement with additional hidden layers. ANN3, ANN4, and ANN5 achieved significantly lower IVRMSE on DOTMC options compared to ANN1. Specifically, ANN5 achieved the lowest IVRMSE for DOTMC options among all the ANN models. Nevertheless, DOTMC options only account for roughly 2% of the options in the test set across all windows, in contrast to ATM options constitutes around 28%. LightGBM and ANN perform better in predicting options with longer time-to-maturity and deep out-the-money options. Intuitively, options with longer time-to-maturity are more stable as they are less affected by short-term fluctuations in the market condition. Additionally, shorter period options have higher theta, which refers to the rate option price changes with respect to time. Although the dynamics change over time, a considerable portion of non-linear relationships in the future could still be captured with ANN and LightGBM trained using historical data. Furthermore, we typically see market conditions as cyclical, thus including more historical data could potentially improve model

fit. Table 10 reports IVRMSE of the models in four different windows for the second exercise. Again, we see relatively large IVRMSE on the 3rd window compared to IVRMSE on the other windows. LightGBM in the second exercise seems to suffer from changing market conditions than LightGBM in the first exercise. Overall, LightGBM and ANN could generalize relatively well compared to Black-Scholes prediction. Overall, LightGBM performs significantly better compared to ANN, achieving lower IVRMSE even when grouping options on different time-to-maturity and moneyness

Table 9: IVRMSE Summary in Second Exercise

	Time-to-maturity			Moneyness					Average
	SHORT	MEDIUM	LONG	DOTMC	OTMC	ATM	OTMP	DOTMP	
LightGBM	0.0215	0.0135	0.0097	0.0210	0.0198	0.0195	0.0169	0.0120	0.0171
ANN1	0.0442	0.0314	0.0249	0.0634	0.0344	0.0317	0.0306	0.0444	0.0370
ANN2	0.0500	0.0473	0.0319	0.0349	0.0365	0.0502	0.0491	0.0503	0.0477
ANN3	0.0541	0.0381	0.0220	0.0399	0.0341	0.0428	0.0386	0.0557	0.0448
ANN4	0.0398	0.0277	0.0186	0.0292	0.0301	0.0333	0.0236	0.0401	0.0329
ANN5	0.0600	0.0426	0.0250	0.0254	0.0378	0.0450	0.0331	0.0696	0.0499
Black-Scholes	0.1114	0.0898	0.0649	0.0865	0.1025	0.0933	0.0502	0.1274	0.0982

This table reports the IVRMSE summary of model predictions considered in the second exercise. SHORT, MEDIUM, and LONG refer to options with time-to-maturity of 1 to 29 days, 30 to 179 days, and 180 days or longer. DOTMC, OTMC, ATM, OTMP, DOTMP refer to options with moneyness [0.8, 0.9), [0.9, 0.97), [0.97, 1.03), [1.03, 1.1), and [1.1, 1.6] respectively. The column Average denotes the average IVRMSE for all options.

Table 10: Comparison of LightGBM and ANN IVRMSE in different windows for the second exercise

	Windows			
	1st	2nd	3rd	4th
LightGBM	0.0098	0.0135	0.0279	0.0096
ANN1	0.0291	0.0324	0.0562	0.0170
ANN2	0.0340	0.0737	0.0306	0.0258
ANN3	0.0219	0.0574	0.0508	0.0345
ANN4	0.0316	0.0383	0.0305	0.0285
ANN5	0.0610	0.0517	0.0477	0.0341

This table reports the IVRMSE summary of LightGBM and ANN models in different windows for the second exercise.

5 Conclusion

In this research, we attempt to predict the implied volatility surface (IVS) using popular contemporary non-parametric approaches. Models in this study include the artificial feed-forward network (ANN), long short-term memory recurrent neural network (LSTM), and gradient boosted decision tree LightGBM. We consider two different approaches. In the first approach, we frame the problem as time series prediction, whereas in the second approach we try to learn the shape of IVS in general. In an attempt to capture time-varying dynamics, we also include Realized Volatility measures with different horizons. Using SPX (S&P 500 Index) European options, we train the models and evaluate their out-of-sample prediction.

We found that LSTM and LightGBM could capture the dynamics of IVS better when a longer sequence of features is included. In LSTM, this is implemented by allowing longer timestep, and in LightGBM lagged observations of the features up to 42 days in the past are included as predictor variables. Based on the result, non-parametric models implemented did not manage to outperform Naive prediction in 1-day and 5-day ahead prediction. However, in 21-day ahead prediction, both LSTM and LightGBM performed favorably against Naive prediction in terms of IVRMSE. Implementing a longer timestep could potentially improve model performance, especially in 21-day ahead prediction. However, this would substantially increase computational time, as in the case of LSTM the complexity scales quadratically with respect to timestep. Another possible improvement could

be including other relevant features that might be excluded in this study. Additionally, as the time series is generated from interpolated IVS, other methods of interpolation can be considered.

In the second approach, we see that ANN and LightGBM show promising capability in learning the Implied Volatility Surface in general. Both models can partially capture the non-linear relationship in the implied volatility surface and could potentially benefit from adding more relevant features, with the downside of additional computational time. Taking into account the first and second exercises, LightGBM shows the most promising result. Particularly, with the setup in the second exercise, LightGBM could predict the Implied Volatility as a function of time-to-maturity and moneyness. Modeling the implied volatility as time series in the setup in this study restricts prediction to a chosen moneyness and time-to-maturity combination. Although, accurate time series prediction of several options with different time-to-maturity and moneyness could give an impression of future the Implied Volatility Skew. Hence, it is possible to use the predictions from the first exercise as features in the second exercise.

Overall, this study shows the ability of non-parametric approaches in predicting IVS, namely ANN, LSTM, and LightGBM. With minimal requirements (none about the data generating process modeling), neural networks and gradient boosted decision trees provide great flexibility in modeling the non-linearity between the Implied Volatility Surface and correlated features. Furthermore, as Almeida et al. (2022) has shown, non-parametric approaches could be augmented with parametric approaches, improving model prediction. Although methods such as neural networks and gradient-boosted decision trees lack interpretability, they offer an uncomplicated approach to modeling non-linear relations. For example, different forecasting method predictions can be included as features in the gradient boosting decision tree method.

References

Y. Ait-Sahalia and A. W. Lo. Nonparametric estimation of state-price densities implicit in financial asset prices. *The Journal of Finance*, 53(2):499–547, 1998. doi: 10.1111/0022-1082.215228.

C. Almeida, J. Fan, F. Tang, and G. Freire. Can a machine correct option pricing models? 2022. working paper.

S. B. Aruoba, F. X. Diebold, and C. Scotti. Real-time measurement of business conditions. *Journal of Business amp; Economic Statistics*, 27(4):417–427, 2009. doi: 10.1198/jbes.2009.07205.

S. R. Baker, N. Bloom, and S. J. Davis. Measuring economic policy uncertainty*. *The Quarterly Journal of Economics*, 131(4):1593–1636, 2016. doi: 10.1093/qje/qjw024.

F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973. doi: 10.1086/260062.

R. Cont and J. da Fonseca. Dynamics of implied volatility surfaces. *Quantitative Finance*, 2(1): 45–60, 2002. doi: 10.1088/1469-7688/2/1/304.

F. Corsi. A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2):174–196, 2008. doi: 10.1093/jjfinec/nbp001.

E. Derman and I. Kani. Riding on a smile. *Risk*, 7(2):32–39, 1994.

B. Dumas, J. Fleming, and R. E. Whaley. Implied volatility functions: Empirical tests. *The Journal of Finance*, 53(6):2059–2106, 1998. doi: 10.1111/0022-1082.00083.

B. Dupire et al. Pricing with a smile. *Risk*, 7(1):18–20, 1994.

S. Goncalves and M. Guidolin. Predictable dynamics in the samp;p 500 index options implied volatility surface*. *The Journal of Business*, 79(3):1591–1635, 2006. doi: 10.1086/500686.

P. S. Hagan, D. Kumar, A. S. Lesniewski, and D. E. Woodward. Managing smile risk. *The Best of Wilmott*, 1:249–296, 2002.

T. Hastie, J. Friedman, and R. Tisbshirani. *The elements of Statistical Learning: Data Mining, Inference, and prediction*. Springer, 2017.

S. L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The review of financial studies*, 6(2):327–343, 1993.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

J. C. Jackwerth and M. Rubinstein. Recovering probability distributions from option prices. *The Journal of Finance*, 51(5):1611–1631, 1996. doi: 10.1111/j.1540-6261.1996.tb05219.x.

G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf`.

H. Y. Kim and C. H. Won. Forecasting the volatility of stock price index: A hybrid model integrating lstm with multiple garch-type models. *Expert Systems with Applications*, 103:25–37, 2018.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

C. Luong and N. Dokuchaev. Forecasting of realised volatility with the random forests algorithm. *Journal of Risk and Financial Management*, 11(4):61, 2018.

M. Rubinstein. Implied binomial trees. *The journal of finance*, 49(3):771–818, 1994.

G. Skiadopoulos, S. Hodges, and L. Clewlow. *Review of Derivatives Research*, 3(3):263–282, 2000. doi: 10.1023/a:1009642705121.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15 (1):1929–1958, 2014.

D. Watson. *Contouring: a guide to the analysis and display of spatial data.* Elsevier, 2013.

J. H. Wright. Options-implied probability density functions for real interest rates. *45th issue (September 2016) of the International Journal of Central Banking*, 2018.

6 Appendix

A LightGBM Hyperparameters

A.1 First Exercise

Table 11: LightGBM Hyperparameter Exercise 1

	h=1			h=5			h=21		
Timestep	3	21	42	3	21	42	3	21	42
Number of Leaves	3184	1136	3184	48	208	2160	2320	2736	1584
Min. Samples in Leaf	15	31	18	86	137	237	84	164	87
Number of Boosted Trees	50	100	800	1450	50	1200	1200	1550	600
Max. Depth	9	8	6	9	6	8	5	12	9
Learning Rate	0.114	0.020	0.024	0.007	0.076	0.109	0.030	0.106	0.099

A.2 Second Exercise

Table 12: LightGBM Hyperparameter Exercise 1

Hyperparameter	Value
Number of Leaves	400
Min. Samples in Leaf	313
Number of Boosted Trees	2000
Max. Depth	11
Learning rate	0.029