International Journal of Data Science and Analytics
https://doi.org/10.1007/s41060-024-00528-7

REGULAR PAPER

Forecasting implied volatilities of currency options with machine learning techniques and econometrics models

Asbjørn Olsen[1] · Gard Djupskås[1] · Petter Eilif de Lange[2] · Morten Risstad[3]

Received: 3 November 2023 / Accepted: 19 February 2024

Abstract
Developing an effective modeling framework to minimize foreign exchange (FX) risk is of vital importance for hedgers and traders in FX markets. In this study, we compare the ability of long short-term memory (LSTM) models to that of random forest and several time series models for forecasting EURUSD implied volatility across the volatility surface. As our literature study argues, there are only a few published papers on this subject. We find that the LSTM model is the best model for shorter option maturities, while the AR-GARCH model is superior when the maturities increase. We observe that the LSTM model is able to capture immense and immediate changes in implied volatility, which is important for hedging against significant shifts in FX rates.

Keywords FX risk · Forecasting implied volatility · Machine learning · LSTM models · Random forest

JEL Classification G17 · G13

1 Introduction

Forecasting market volatility including implied volatility of foreign exchange (FX) options is challenging mainly due to incomplete information and unprecedented changes in economic trends and conditions. Volatility forecasts are utilized by volatility traders seeking alpha and institutional fund managers such as life insurers and pension funds for hedging purposes. Also, central banks need to have a forward-looking view on exchange rates, because changes in exchange rates impact inflation in open economies. Improving volatility forecasts is thus an important task. Traditional econometric time series models struggle to capture nonlinearity in data, incentivizing economic researchers to utilize more advanced models. Machine learning methods can alleviate the complexity in time series forecasting by identifying structures and patterns of data such as nonlinearity and dependency between predictors. Particularly, LSTM (long short-term memory) networks have received increased focus in forecasting financial time series due to their specific attributes.

Developing an effective modeling framework to minimize FX risk is of vital importance for hedgers and traders in FX markets. The subject of this study is to compare the predictive power of LSTM models to random forest and time series models for forecasting implied volatility of options on the EUR/USD foreign exchange rate. We compare the ability of long short-term memory (LSTM) models to that of random forest, AR-GARCH, HAR and MIDAS models for forecasting EURUSD implied volatility across the volatility surface. The LSTM model (or network) is a type of recurrent neural network (RNN). We find that the LSTM model is the best model for shorter option maturities, while the AR-GARCH model is superior when the maturities increase. However, imposing other specifications and residual distributions for the GARCH models, we find that the AR-GARCH

✉ Petter Eilif de Lange
petter.e.delange@ntnu.no

Asbjørn Olsen
asbjornolsen94@gmail.com

Gard Djupskås
gard_djupskas@hotmail.com

Morten Risstad
morten.risstad@ntnu.no

[1] Department of Economics, Norwegian University of Science and Technology, Trondheim, Norway

[2] Department of International Business, Norwegian University of Science and Technology, Trondheim, Norway

[3] Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway

Published online: 25 March 2024

framework outperforms the more advanced machine learning models for all options. We observe that the LSTM network is able to capture immense and immediate changes in implied volatility, which is important for hedging against significant shifts in FX rates.

Reasons for our interest in exploring forecasting capabilities of LSTM models include:

- The LSTM network is designed to deal with a major problem of classical RNNs, the vanishing gradient problem (see Sect. 4.3).
- Its long- and short-term memory property enables the LSTM model to remember volatility clustering, making it well suited to forecasting implied volatility.
- As our literature study below demonstrates, the LSTM network has been successfully applied to forecasting financial time series.

We structure our analysis into two parts:

1. *Statistical distribution* Analyzing the univariate time series characteristics of the implied volatility components, including dependency structure.
2. *Forecasting models* Evaluating and proposing forecasting models for implied volatility.

We use a comprehensive dataset of daily implied volatilities for maturities from one week through one year, covering a broad range of strike levels. The dataset spans important global macroeconomic events such as the financial crisis, the euro sovereign debt crisis and the COVID-19 outbreak.

We optimize an LSTM model on a training set of the data and compare its forecasting capability to a RF (random forest) model and AR-GARCH-type models. The estimator is the daily spot rates for the implied volatility for the EUR/USD FX options. The four most popular currency pairs in terms of trading volume are: EUR/USD (Euro Dollar), USD/JPY (Dollar Yen), GBP/USD (Pund Dollar), USD/CHF (Dollar Swiss Franc). These crosses are highly correlated. In this study we decided to explore the EUR/USD because it is the most important and liquid currency pair.

We impose RF and a Gaussian distributed AR(1)-GARCH(1,1) as benchmark models and compare their forecasting performance to the more advanced LSTM model. In addition to the benchmark AR(1)-GARCH(1,1) model, we extend the analysis with models that include an asymmetric GARCH term, moving average terms, along with Gaussian distributed residuals and Student's t-distributed residuals. The machine learning models are optimized using different hyperparameters, and we compare the best-fitted structures for each option to the benchmark models.

Our findings show that the LSTM model is better than the benchmark models for shorter option maturities, while the AR-GARCH model is superior when the maturities increase. However, when imposing other specifications and residual distributions for the GARCH models, we find that the AR-GARCH framework outperforms the more advanced machine learning models for all options. For shorter maturities the t-distributed models perform best, while ARIMA-GARCH-type models perform better for longer maturities.

Further, this study is organized as follows: Sect. 2 discusses previous publications on implied volatility and the models we apply in our analysis. Section 3 presents the data, including statistical and distributional behavior. Section 4 presents the theory behind our models and further describes the methodology and model architecture. In Sect. 5, we present results and findings. Section 6 summarizes our findings and concludes.

2 Literature review

Garman and Kohlhagen (1983) derive an implied volatility modification to the Black–Scholes formula for option pricing, introduced by Fischer Black and Myron Scholes [3]. According to Stan (1981), Latane and Rendleman (1976), several studies have shown that implied volatility (which is utilized in this study) is a better forecaster of future price variability than measurements based on history [2]. In recent years, implied volatility has become a common estimator for forecasting purposes.

Ornelas and Mauad [31] find that the slopes of currency implied volatility term structures have predictive power for the behavior of exchange rates from both cross-sectional and time series perspectives. Carr et al. [8] build a volatility index by formulating a variance prediction model using machine learning methods such as feedforward neural networks and random forest on the S&P 500 index options. According to Haug et al. [17], the standard deviation of implied volatility has an evident variation over time and declines as time to maturity increases. We observe the same phenomenon (see Fig. 2). Time-varying properties entail a major challenge, volatility clustering. That is, small (big) changes in the volatility tend to be followed by small (big) changes in the volatility [26].

An attempt to account for volatility clustering is Robert Engle's nonlinear autoregressive conditional heteroskedasticity (ARCH) model, allowing the time-varying conditional variance to depend on the lagged values of the squared errors. An extension to the ARCH model that allows the conditional variance to depend on lags of the conditional variance is the general ARCH model, or the GARCH model, introduced by Bollerslev [4]. The GARCH model is more parsimonious than the ARCH model. The GARCH model avoids overfitting and is still today a much-applied modeling framework

for financial time series data. In this study, we employ random forest and different time series models as benchmarks (see Sect. 4.2).

Glosten et al. [15] formulate an extension to the GARCH model, which accounts for an asymmetric response to a volatility shock. In this model, known as the GJR-GARCH, "good" news and "bad" news have different impacts on the subsequent period volatility. Lim and Sek [24] found that in "normal" post- and pre-crisis times, the symmetric GARCH performs well, and in times of big volatility fluctuations, i.e., times of crisis, the asymmetric model is preferred. Schmidt [36] argues that the asymmetric models are better forecasters for financial indexes in the aftermath of the shock caused by the outbreak of the COVID-19 pandemic compared to symmetric specifications. Poon and Granger [34] argue that the simpler GARCH models seem to provide larger volatility forecasts compared to the more sophisticated models. In contrast, the GJR-GARCH seems to forecast lower values due to its asymmetry, which helps this model to quickly revert from a high volatility state to a low volatility state. Ramasamy and Minusamy [35] found that the asymmetric GJR does not improve the forecasting performance considerably compared to symmetric GARCH models.

The literature dedicated to implementing machine learning techniques for forecasting the implied volatility of FX options is scarce. However, some research exists studying machine learning for predicting stock prices, returns and volatility.

Hosker et al. [16] compare three existing financial models that forecast future market volatility using the Chicago Board Options Exchange Volatility Index (VIX) to six machine/deep learning supervised regression methods. They discover that RNNs including LSTM (which is used in our study) provide improved results over existing linear regression, principal components analysis (PCA) and ARIMA methods. Medvedev and Wang [28] model the implied volatility surface (IVS) using convolutional long short-term memory (ConvLSTM) and long short-term memory (LSTM) neural networks to produce multivariate and multistep forecasts of the S&P 500 IVS. They conclude that the ConvLSTM model significantly outperforms LSTM and traditional time series models in predicting the IVS out of sample.

Galler and Kruzanowski [22] employ deep learning to classify whether stock returns are positive or negative one-year-ahead. More recently, Krauss et al. [23] use various machine learning models, such as deep learning and tree-based methods, to model S&P 500 constituents. Surprisingly, Krauss et al. [23] reported that gradient-boosted trees and random forest outperformed deep learning models. Interestingly, Krauss et al. [23] revealed that deep learning models perform exceptionally well in times of market turmoil. Also, Yu and LI's (2018) findings are consistent with the claim that deep learning networks perform well during market turmoil. Yu and Li (2018) forecast the volatility of the Shanghai composite stock price index using LSTM and GARCH, selecting only extreme values (highs and lows) and concluding that the LSTM model is superior. A paper somewhat similar to ours, which forecasts different stock indices, is Namin and Namini [29]. They compare an Arima model to a univariate multistep LSTM model, employing a multi- step univariate forecasting algorithm based on Brownlee [7]. They conclude that the LSTM model outperforms the ARIMA model. Galakis and Vrontos [38] study whether the application of machine learning approaches can outperform traditional econometric models in forecasting implied volatility indices, a task similar to ours. They conclude that certain machine learning techniques are strongly encouraged as they significantly improve the accuracy of the out-of-sample forecasts. However, they also report that the model accuracy is not consistent across all models.

3 Data: distribution and statistical behavior

In this Section, we discuss the statistical properties and distribution of our data. Our data, retrieved from Bloomberg, consist of daily observations of implied volatilities for eleven options with distinct levels of moneyness and five different times to maturity during the period 02.01.2007–31.08.2021. The data set provides 55 distinct time series with 164.670 observations of implied volatility, enabling us to analyze our models' forecasting performance for different maturities and distinct moneyness levels. Figure 1 and Table 1 summarize descriptive statistics for ATM (at-the-money) put options for the five distinct maturities. The variance of the volatilities generally declines as time to maturity increases. The shorter maturities have both higher peaks and lower troughs of implied volatility. In comparison, the longer maturities have higher average levels of implied volatility, measured in both mean and median (50% quantile).

In this study, we use put and call options with OTM delta values of 5, 10, 18, 25, 35 and ATM put options with a delta of 50. The level of implied volatility, measured in mean and at different quantiles, is higher for options OTM than ATM or close to ATM, and it is higher for puts than for calls. This is also the case for the volatility (i.e., daily changes in the level of implied volatility). This distribution pattern is the same for all five distinct maturities and is referred to as the volatility smile, which is visualized in Fig. 2. The implied volatility is higher for OTM put options than similar call options, consistent with a negative risk reversal that measures the volatility smile's skewness. It is most common to measure the risk reversal for call and put options with a delta of 25 [27]. On average, all 25-delta risk reversals are negative and increasingly negative as the time to maturity increases. Further, the risk reversals become increasingly negative as the

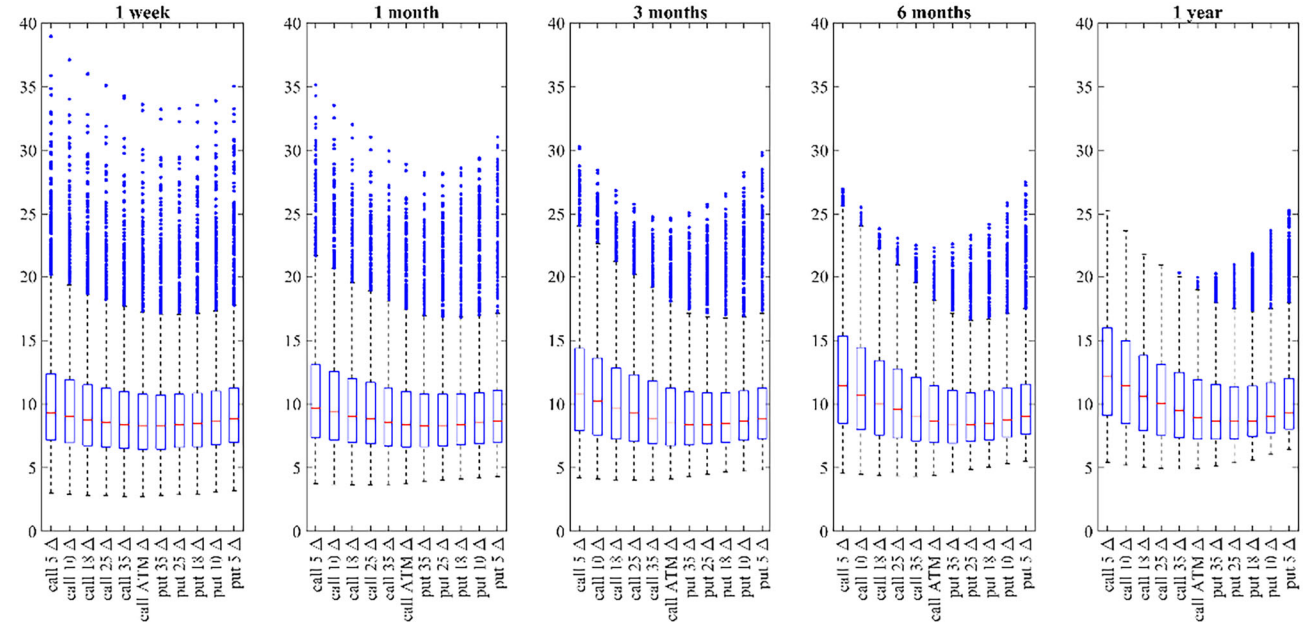

Fig. 1 Empirical distributions of implied volatilities across maturities and levels of moneyness. Level of implied volatility along the vertical axis and level of moneyness along the horizontal axis

Table 1 Descriptive statistics for ATM put options for each maturity

	1 week	1 month	3 months	6 months	1 year
Obs	2994	2994	2994	2994	2994
Mean	9.14	9.16	9.29	9.43	9.62
Min	2.74	3.77	4.14	4.42	4.97
25%	6.43	6.64	6.75	6.96	7.23
50%	8.32	8.38	8.51	8.63	8.95
75%	10.76	10.96	11.24	11.44	11.91
Max	33.58	28.88	24.65	22.29	19.91
Var	14.70	12.58	11.26	10.28	9.32
σ	3.83	3.55	3.36	3.21	3.05

options become increasingly OTM and can be interpreted as a market-based measure of implied skewness.

Comparing historical values of the implied volatility for one-week and one-year options (see Fig. 3), the variation in implied volatility is immense. Since the one-week option has a shorter time to maturity, implied volatility reacts more to news and small shocks and is more volatile. The cost of short-term options is lower than that of long-term options, which results in higher demand for short-term options. This causes longer maturity options to trend more, recovering less rapidly from massive shocks than shorter maturities, implying that time series for longer maturities are non-stationary. As time to maturity increases, we observe a decline in how often the option crosses its mean. The fact that the amplitude of the daily changes in implied volatility declines will contribute to a change in the tails of the return distribution. Since there is a significant difference in the behavior and distribution of the different maturities and levels of moneyness, we expect that different models will be the best fit for different options. We will come back to this in Sect. 4.

Several macroeconomic factors impact the EUR/USD exchange rate, and its implied volatility. Our data set stretches from January 2007 to August 2021 and during this time, the financial markets worldwide endured multiple shocks and events that impacted the EUR/USD exchange rate. Figure 3 exhibits the implied volatility for the ATM one-week and one-year options. Shortly following the financial crisis in 2008, we observe a considerable rise in implied volatility. Also, shocks such as the European debt crisis, the US debt ceiling crisis, and in more recent years, the COVID-19 pandemic had a significant impact on implied volatility. These different shocks captured in the data will affect the implied

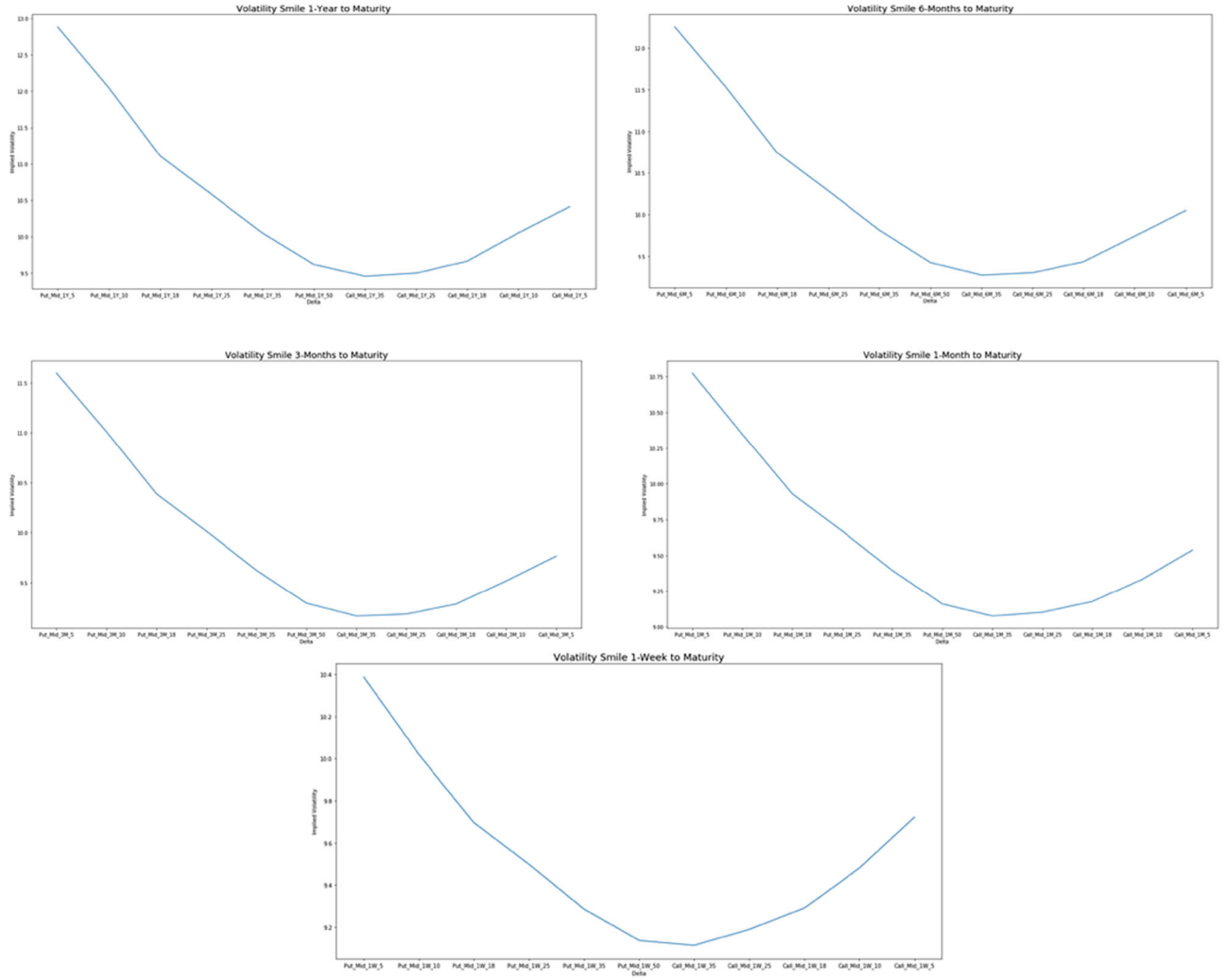

Fig. 2 Volatility smiles for the different maturities. Volatility smiles for each distinct maturity, calculated as an average across the data sample. The top left option has one year to maturity, the top right has six months, mid left has three months, mid right has one month and the bottom option has one week to maturity. Level of implied volatility is measured along the vertical axis, and level of moneyness along the horizontal axis

volatility, the daily return of the implied volatility, and the residual distribution of the options. Presumable, this can be a challenge for GARCH-type models, which assume normally distributed residuals. Outliers for the shorter maturity options result in fatter tails than would be observed in a normal distribution. We will further address these issues in Sect. 5.5.

Different shocks to the data produce positive skewness for all option maturities, as illustrated by the right-tailed empirical distribution of the ATM options in Fig. 4. The figure exhibits the ATM option for each maturity, and the distribution widens as time to maturity increases. The empirical distribution has a double peak for the longest maturities, caused by more extended periods away from their respective mean value, as visualized in Fig. 3. The distribution pattern corresponds to OTM options, and our findings regarding EUR/USD FX derivatives' statistical and distributional behavior align with earlier literature (Figs. 5, 6).

4 Methodology

4.1 Forecast evaluation

To evaluate the forecasting performance of the different models we use mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE). These statistical measurements are given by the formulas:

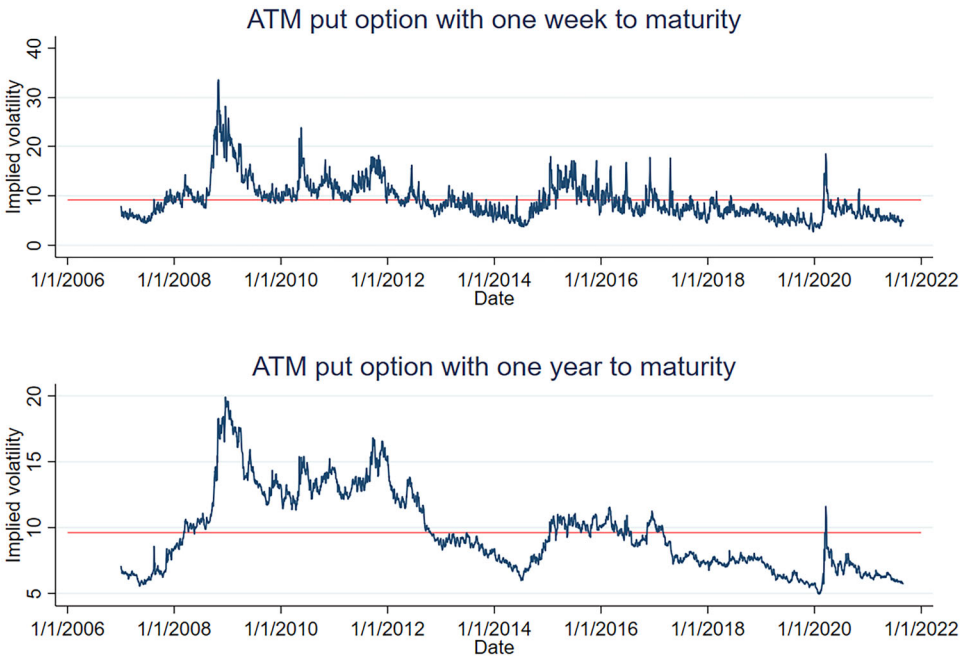

Fig. 3 Implied volatility for ATM put options with one week and one year to maturity. Spot rates of implied volatility for ATM put options with one week and one year to maturity from 2. January 2007 until 31. August 2021. The red horizontal line exhibits the options mean value for the sample period

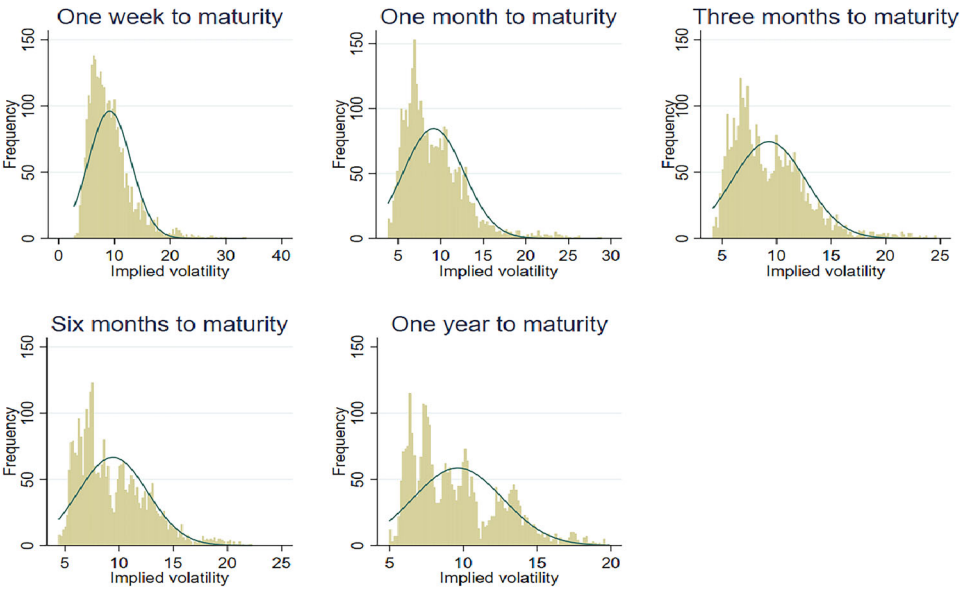

Fig. 4 Empirical distribution for ATM options for each maturity, Gaussian distribution drawn for each maturity. Empirical distribution of ATM options for each of the five maturities. The distributions implied volatility along the horizontal axis, and frequency measured in number of observations along the vertical axis. The Gaussian distribution is marked as a curved line for each plot. There are 200 bins and 2994 observations for each maturity plot

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 \tag{1}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2} \tag{2}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}|$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}|/y_i \tag{3}$$

where y_t is the implied volatility at time t, $\widehat{y_i}$ is the forecasted value of implied volatility at time t, and n is the number of observations. While the mean absolute error measures the average across errors, where the errors are weighted equally, the mean squared error penalizes higher errors.

4.2 Benchmark models

We use GARCH and supervised random forest as our benchmark models. Common practice is to use 80% of the dataset as a training set and 20% as an out-of-sample test set. This

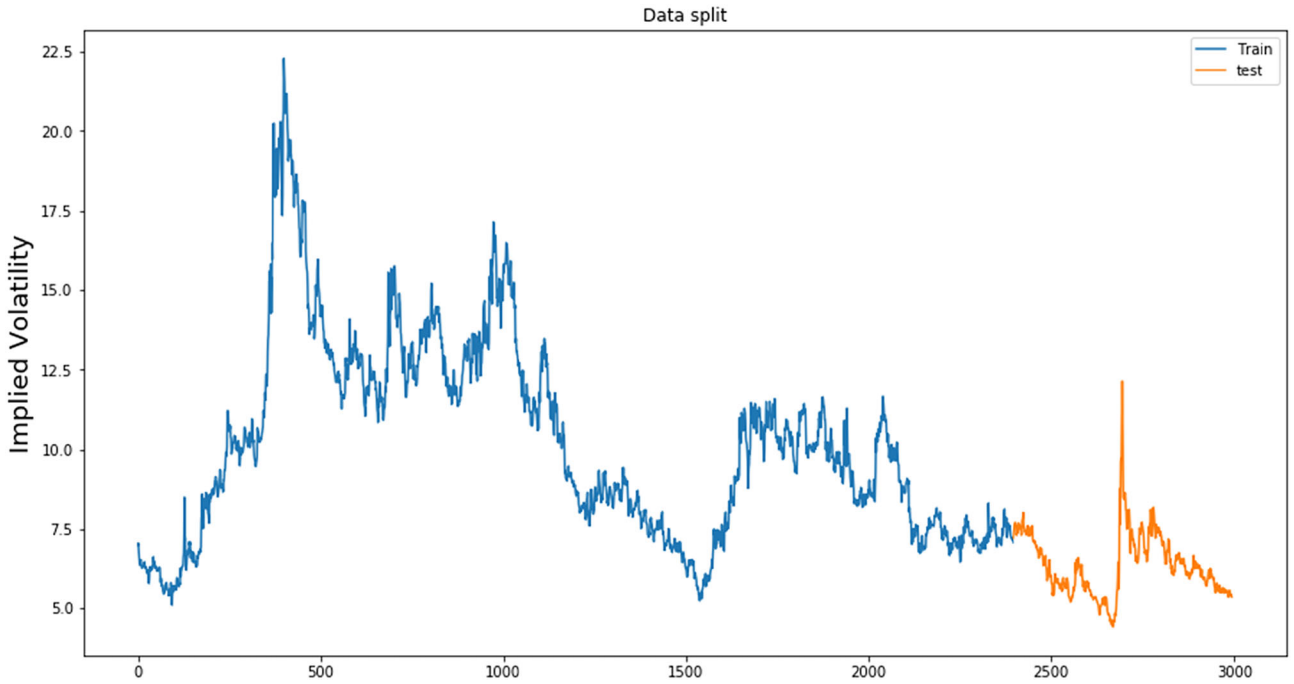

Fig. 5 80:20 data split for training and test set, option exhibited is ATM put option with six months to maturity. 80:20 data split for an ATM put option with six months to maturity

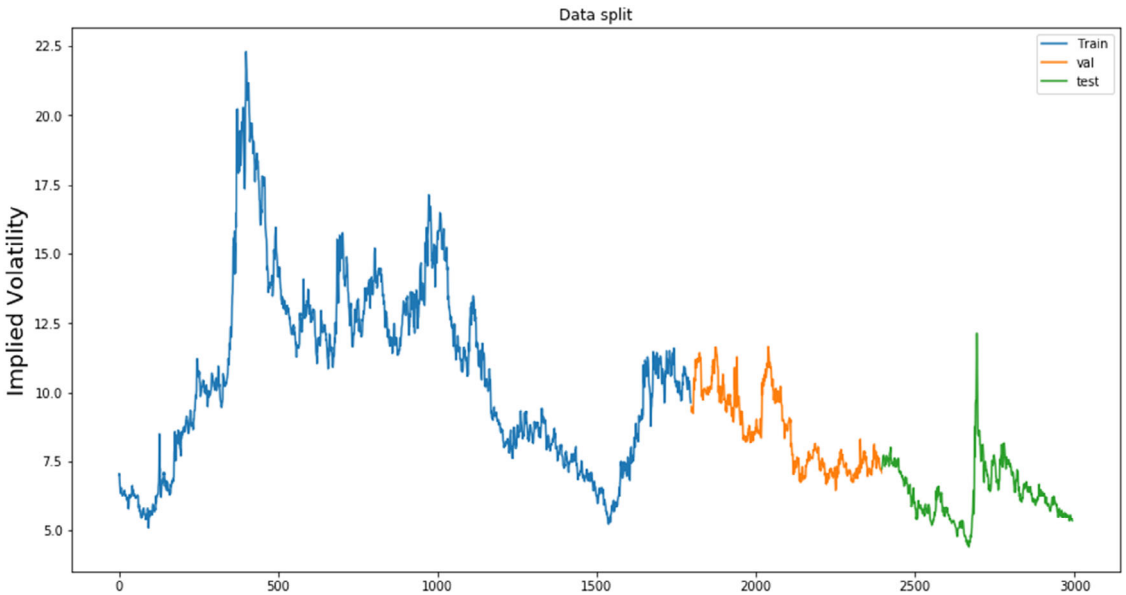

Fig. 6 Data split for training, validation and test set for LSTM model. LSTM Data sets split into training, validation and test sets

split produces 2396 observations for the training set and 599 for the out-of-sample test set.

4.2.1 AR-GARCH

To confirm the presence of autoregressive conditional heteroskedasticity in the data, we perform Engle's Lagrange multiplier test, also known as an ARCH-LM test. The results from the ARCH-LM test can only be interpreted as an indication of whether ARCH effects are present or not, and according to Sjölander [37], the test is biased in finite samples. It does not consider whether the stationarity constraints are met. Test results show that for all options, we can at any significance level reject the null hypothesis of no ARCH effects. We conclude that there is evidence of autoregressive

conditional heteroskedasticity in the squared residuals for all options.

As our literature study revealed, autoregressive time series with random disturbances displaying conditional heteroscedastic variances have been successfully modeled with GARCH-type processes. In this study, we employ an AR(1)-GARCH(1,1) model as our benchmark. We extend the analysis with a GJR-GARCH specification to capture asymmetric behavior of shocks in the conditional variance. We compare both in-sample goodness of fit and out-of-sample forecast errors for all models, emphasizing the out-of-sample forecasting performance. Further, we examine the data with additional lags of the AR(p) and MA(q) processes, and lastly, we perform the same regressions with a Student's t-distribution. Additional lags beyond an ARMA(1,1) process do not improve the model's forecasting accuracy and will not be included further in this study.

Non-stationary variables can produce spurious regressions, yielding a high R^2 and t-statistics which appear to be significant but without any economic meaning [11]. To test for stationarity, we use the augmented Dickey-Fuller test. Based on the augmented Dickey-Fuller test and the Phillips-Perron test, we cannot, at a 5% significance level, reject the null hypothesis that the options with time to maturity of 3 months and more follow unit root processes and are non-stationary. To avoid spurious regressions, we apply first difference to all options with three months or more to maturity. When applying the first difference for the options with one week and one month to maturity, the forecast errors measured by MSE, RMSE and MAE are reduced, along with the in-sample goodness of fit. Therefore, the benchmark AR(1)-GARCH(1,1) model is not differentiated for the shorter maturities, i.e., one week and one month to maturity.

The in-sample goodness of fit for the models are compared by the information criteria Log-Likelihood, AIC and BIC. According to these information criteria, the in-sample model is considerably improved for all distinct options when adding a moving average (MA) term and more autoregressive lagged terms. In terms of the autocorrelation and partial autocorrelation, there are individual preferences of which lags of AR(p) and MA(q) terms should be included, for the different options. Testing for asymmetries in the conditional variance, the threshold term improves the goodness of fit, and is statistically significant, for all options across the level of moneyness and maturities. Based on these findings, we can claim that there is asymmetry in the volatility shocks, i.e., the volatility reacts differently to positive and negative shocks. In terms of out-of-sample forecasting accuracy measured by MSE, RMSE and MAE, the simple AR(1)-GARCH(1,1) model outperformed the better in-sample specified models on shorter maturities. Interestingly, the better the in-sample model specification measured in goodness of fit, the poorer is the out-of-sample forecasting accuracy for the short maturity options. Still, on account of its simplicity, we use the AR(1)-GARCH(1,1) as a benchmark model.

4.2.2 Random forest

Random forest model architecture Developing our model architecture, we implement the random forest regressor from scikit learn [33]. The sklearn.ensemble.RandomForestRegressor has several parameters that can be tuned. Each time a split is to happen in a decision tree, a random sample of m predictors are chosen from a total of p predictors. In Python's sklearn, the default number of features m used when making splits in a random forest regression, equals number of predictors in the regression problem p. Breiman [6] argued that the optimal number of features should be the square root of p. Hastie et al. [18] argued that p / 3 is the set of features best suited for the random forest regressor. From empirical investigation Geurts et al. [12] concluded that the optimal set of features is simply $m = p$. Running our random forest model for different sets of m on different options, we reach the same conclusion that $m = p$ (see Table 2).

Next, we examine the number of trees and the length of the window. Our initial model employed ten trees and a window size of two. Increasing the number of trees to the sklearn default option of one hundred, did reduce the mean squared error significantly for options with a very short time to maturity. We also ran the model with window sizes from two to forty. However, this was computationally too expensive to do for each option. Table 3 depicts the mean squared error for an ATM option with one year to maturity. The table indicates that increments in window size do not improve the mean squared error.

4.2.3 HAR

The HAR framework introduced by Corsi [10] has repeatedly proved its relevance for volatility forecasting, in spite of a relatively simple structure. The original specification proposed in Corsi [10] uses lagged values of the dependent variable as covariates:

$$\widehat{y_i} = \mu + \beta_D y_{t-1} + \beta_W y_{t-5} + \beta_M y_{t-22} \tag{4}$$

More specifically Eq. (4) uses a daily, a weekly and a monthly lag of implied volatility with a given moneyness and time to expiry, along with corresponding coefficients, to forecast implied volatility. Equation (4) is a linear model and can be estimated using ordinary least squares.

Table 2 Search for optimal features in the RF model for ATM put option with one year to maturity

Features	$m = p$	Log2 (p)	Sqrt (p)	$p/3$
One Year ATM Put	0.0404	0.0448	0.0448	0.0444

Search for optimal features. We selected $m = p$ for our RF model

Table 3 Search for optimal window size RF model for ATM put option with one year to maturity

	2	10	20	30	40
Window size					
One Year ATM Put	0.0404	0.0421	0.0412	0.0426	0.0412

Search for Optimal Window Size. Increasing the window size has negligible effect on MSE

4.2.4 MIDAS

MIDAS regressions [13] are essentially tightly parameterized, reduced form regressions that accommodate processes sampled at different frequencies and with different lags. The response to the higher-frequency explanatory variable relationship between dependent and independent variables is modeled using highly parsimonious distributed lag polynomials.

The basic MIDAS model for h-step-ahead forecasting, with data available up to x_t, is given by:

$$y_{t+h} = a_h + b_h C(L; \theta_h) x_t + \varepsilon_{t+h} \tag{5}$$

where $C(L; \theta) = \sum_{i=0}^{N} c(i; \theta) L$ and $C(1; \theta) = \sum_{j=0}^{N} c(j; \theta) = 1$. In our application, x_t contains L = 22 lagged values of implied values.

Parsimonious parameterization of the lagged coefficients of $c(k; \theta)$ is one of the key MIDAS features. Various specifications for $C(L; \theta)$ exist (see Ghysels and Marcelliano [14] for a discussion). One alternative, which we apply in this study, are Legendre polynomials as proposed by Babii et al. [1]:

$$b_h c(i; \theta = [\theta_0, \ldots, \theta_P]) = \sum_{p=0}^{P} \theta_p L_p(x_i)$$

where the Legendre polynomial of order $p = 3$ takes the form: $L_0(x) = 1$, $L_1(x) = x$, $L_2(x) = \frac{3}{2}x^2 - \frac{1}{2}$, $L_3(x) = \frac{5}{2}x^3 - \frac{3}{2}x$.

4.3 Recurrent neural networks

The recurrent neural network is a class of artificial neural networks, which are software implementations of the network of neurons present in the human brain. The RNN is a network where neurons have "memory," which can capture temporal dependencies in the data, that can be modeled in a recurrent structure, Ni et al. [30]. When training RNNs with gradient-based learning methods and backpropagation, the gradient of the error function may approach zero "too" fast terminating the training process (vanishing gradient) or explode. This means that RNNs are unable to remember *long*-term dependencies. The LSTM has been designed to relieve the vanishing gradient problem of RNNs, thus keeping track of long-term dependencies in the input sequences (the sample time series). The long- and short-term memory property of the LSTM network presumably enables it to remember volatility clustering patterns in the distant and near past, making this model particularly well fit to forecasting implied volatility.

Below we derive the architecture for our LSTM model.

4.3.1 LSTM data split

First, we split our dataset into training, validation and test sets. This is important as the model will be near perfect if we feed the model with the test data. We settle on a 60:20:20 split, where 60% is used as the training set, 20% is used as the validation set and 20% is used as a test set. In other words, the first 1797 of the first observations in the dataset are used to train the model, the next 599 observations are used to improve the model and fine-tune hyperparameters and the last 599 observations are used for out-of-sample forecasts.

The date intervals are as follows:

- Training set: January 2007–October 2015
- Validation set: October 2015–September 2018
- Test set: September 2018–August 2021

4.3.2 LSTM model architecture and hyperparameter search

Neural network algorithms are stochastic, i.e., they make use of randomness, such as initializing random weights, which will yield different results for a network that is trained on the same data. To improve the LSTM model, we use a random

seed, which generates a long sequence of numbers, which will function as weights in the stochastic algorithm, and ensure that the same result occurs when we run the same model twice.

Several parameters require tuning to optimize the LSTM model. The common practice is to evaluate every possible combination of parameters on the validation set and choose the varieties that minimize the evaluation metrics. However, this approach becomes computationally expensive, especially with an increased window size. For this reason, we develop an architecture that starts by combining smaller sets of hyperparameters, and for each iteration, the hyperparameters increase by 10.

We implement an LSTM model from the Keras functional API [9] with two hidden layers searching for hyperparameters. We tried using different stacks of LSTM layers, which was computationally expensive and did not improve the model. The ReLU activation function is popular because of its computability efficiency [20], but we employ the Keras LSTM built-in activation function tanh, since this function seems to work better for our datasets. We use the sigmoid as the recurrent activation function, and Adam as our optimizer. Adam is a variant of the mini-batch gradient descent algorithm that adjusts the learning rate at each iteration for each model parameter [9]. Our model is specified to minimize the mean squared error. Initially, we construct our model architecture with 300 epochs, a batch size of 64, a window size of 50 and 50 hidden neurons. Another issue with LSTM is overfitting. With excessive training, the model will learn the statistical noise in the training set, predicting the next value based on memory. To avoid overfitting we use early stopping, which terminates training as the learning rate stops improving.[1]

The remaining hyperparameters that need tuning are: batch size, hidden neurons and window size.

4.3.3 Batch size

We use the previously stated model architecture to speed up the hyperparameter search to locate the optimal batch size. Our goal is to find a batch size that minimizes the mean squared error, and the common practice is to increase the batch size by the power of two because of computational efficiencies [21]. Figure 7 shows the results for the batch sizes and indicates an optimal batch size of 16.

[1] In addition to early stopping, we tried implementing Keras dropout [9] and Keras Gaussian noise [9], both techniques aimed at preventing overfitting. However, both features either increased—or unaffected the error statistics.

4.3.4 Combinations of window sizes and neurons

Further, we investigate the combinations of window sizes and hidden neurons, which are the parameters that largely affect the model's ability to learn. We implement the optimal batch size of 16 in our initial model and develop an architecture that combines the different neurons and window sizes from two to fifty. Table 4 reports the mean squared errors for each combination for an ATM option with six months to maturity. The hyperparameter search indicates that a simple model of two lags and twenty hidden neurons minimizes the mean squared error for this specific option.

5 Forecasting results

For the out-of-sample forecast, forecasting accuracy generally increases with moneyness and time to maturity. From one week to maturity to one year to maturity, the average RMSE decreases from 0.7357 to 0.2417 for the GARCH model, from 0.7303 to 0.2521 for the LSTM and from 0.8224 to 0.2473 for the Random Forest. This is a decline of, respectively, 67.16%, 65.47% and 69.92%. The daily change in implied volatility, computed as the absolute value of the average daily change for each maturity, declines by 76.21% when the maturity increases from one week to one year. The reduction in RMSE is therefore declining with longer maturities as expected a priori. The daily change in implied volatility is also lower for options ATM and options close to ATM than for OTM options. It is also lower for call options than put options with the same option delta (negative risk reversal).

5.1 ATM and OTM options summary

The results for an ATM put and an OTM put and call for each of the five specific maturities are presented in Table 5. The first column indicates the options level of moneyness and time to maturity, then forecasting accuracy results for the AR(1)-GARCH(1,1), LSTM, RF, HAR, and MIDAS models follow.

In line with Galakis and Vrontos [38], we find that certain machine learning techniques significantly improve the accuracy of the out-of-sample forecasts, but the results are somewhat mixed. The benchmark AR-GARCH model is performing superior for both put and call options compared to the machine learning methods for options with longer maturities. For all options with maturities of three months or more, the AR-GARCH model outperforms both the machine learning models and other time series models. The random forest model has the lowest forecast performance of the three models for all options across the different maturities. On shorter maturities, i.e., for options with one week and one month to

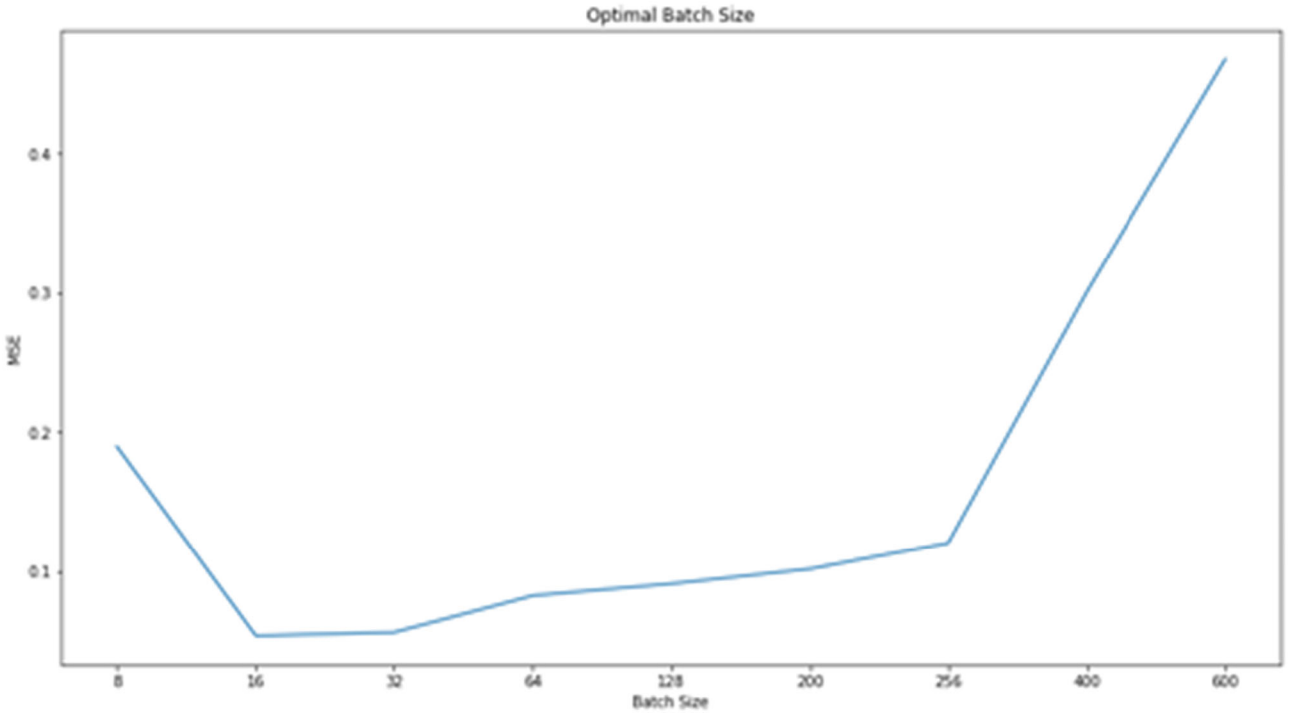

Fig. 7 Optimal batch size for LSTM network on ATM six months of option. Optimal batch size for a Delta 50 put option with six months to maturity

Table 4 Mean squared error estimates for a delta 50 put option with six months to maturity

Neurons	2	10	20	30	40	50
Window size						
2	0.2471	0.0566	0.0831	0.0604	0.0880	0.0597
10	0.0933	0.1121	0.1127	0.2031	0.0609	0.0769
20	**0.0550**	0.1008	0.0800	0.5671	0.0721	0.0589
30	0.1035	0.0824	0.0637	0.0556	0.0611	0.0565
40	0.0801	0.0740	0.0652	0.0579	0.0625	0.0565
50	0.0765	0.0682	0.0639	0.0555	0.0594	0.0555

Reported error estimates for different choices of neurons and window size. The number of neurons along the vertical axis and window size along the horizontal axis. Optimal combination of neurons and window size are highlighted, and for this particular option the best combination of window size and neurons is 20 neurons and a window size of 2

maturity, the LSTM and AR-GARCH models are the best-fitted models. For an OTM put option with one week to maturity, the GARCH model is the best fitted with an RMSE of 0.7913 while the LSTM model has an RMSE of 0.7976. The random forest model performs significantly poorer with an RMSE of 0.9098, an increase of 14.97% compared to the GARCH model.

It is somewhat surprising that the GARCH outperforms the LSTM model for this particular option, considering that this is the most volatile of the fifty-five options. The LSTM performs better in terms of MAE, meaning it is not as robust to outliers in the test set as is the GARCH model. When we perform a Diebold-Mariano (DM) test for this option we see that there are no significant differences between LSTM and GARCH or LSTM and RF. However, the AR-GARCH is significantly better than the RF. According to the DM test, AR-GARCH is significantly better than RF for one-week options. AR-GARCH is significantly better than LSTM for the ATM option, but not for OTM options expiring in one week.

Although the HAR and MIDAS models in certain instances have the highest accuracy, these models are generally not able to improve predictions beyond GARCH and LSTM.

Table 5 Forecasting performance for OTM put/call options and ATM put options

	GARCH	LSTM	RF	HAR	MIDAS
MSE					
1 week put 5	**0.6262**	0.6362	0.8277	0.6374	0.6654
1 week put 50	0.4895	**0.4763**	0.6944	0.4944	0.4951
1 week call 5	0.5947	**0.5917**	0.7820	0.5946	0.6192
1 month put 5	**0.2561**	0.2645	0.2670	0.2599	0.2605
1 month put 50	**0.1500**	0.1537	0.1842	0.1524	0.1508
1 month call 5	0.2519	0.2519	0.2806	0.2487	**0.2480**
3 months put 5	**0.1706**	0.1792	0.1777	0.1782	0.1795
3 months put 50	**0.0773**	0.0850	0.0888	0.0788	0.0797
3 months call 5	0.1387	0.1413	0.1642	**0.1385**	0.1410
6 months put 5	**0.1324**	0.1392	0.1443	0.1419	0.1416
6 months put 50	**0.0506**	0.0550	0.0634	0.0519	0.0522
6 months call 5	**0.0975**	0.1022	0.1137	0.0978	0.0987
1 year put 5	**0.1135**	0.1215	0.1399	0.1228	0.1226
1 year put 50	**0.0367**	0.0398	0.0404	0.0380	0.0377
1 year call 5	**0.0770**	0.0815	0.0841	0.0776	0.0779
RMSE					
1 week put 5	**0.7913**	0.7976	0.9098	0.7984	0.8157
1 week put 50	0.6997	**0.6901**	0.8333	0.7032	0.7036
1 week call 5	0.7712	**0.7692**	0.8843	0.7711	0.7869
1 month put 5	**0.5061**	0.5143	0.5167	0.5098	0.5100
1 month put 50	**0.3873**	0.3920	0.4292	0.3904	0.3883
1 month call 5	0.5019	0.5019	0.5297	0.4987	**0.4980**
3 months put 5	**0.4131**	0.4233	0.4215	0.4221	0.4237
3 months put 50	**0.2781**	0.2915	0.2980	0.2807	0.2824
3 months call 5	0.3725	0.3759	0.4052	**0.3722**	0.3755
6 months put 5	**0.3639**	0.3731	0.3799	0.3767	0.3763
6 months put 50	**0.2248**	0.2345	0.2518	0.2278	0.2284
6 months call 5	**0.3123**	0.3197	0.3372	0.3128	0.3142
1 year put 5	**0.3369**	0.3486	0.3740	0.3504	0.3501
1 year put 50	**0.1916**	0.1995	0.2010	0.1950	0.1943
1 year call 5	**0.2776**	0.2855	0.2900	0.2786	0.2791
MAE					
1 week put 5	**0.5056**	0.4974	0.5973	0.5208	0.5108
1 week put 50	**0.4632**	0.4475	0.5492	0.4692	0.4582
1 week call 5	**0.4813**	0.2833	0.5583	0.4836	0.4834
1 month put 5	**0.2729**	0.2767	0.3055	0.2853	0.2821
1 month put 50	**0.2709**	0.2308	0.2640	0.2351	0.2308
1 month call 5	**0.2518**	0.2513	0.2782	0.2542	0.2598
3 months put 5	**0.2002**	0.2056	0.2346	0.2121	0.2123
3 months put 50	**0.1540**	0.1610	0.1757	0.1621	0.1605
3 months call 5	**0.1772**	0.2278	0.1999	0.1824	0.1822
6 months put 5	**0.1641**	0.1679	0.1963	0.1789	0.1774
6 months put 50	**0.1191**	0.1269	0.1445	0.1197	0.1250
6 months call 5	**0.1433**	0.1449	0.1649	0.1484	0.1467

Table 5 (continued)

	GARCH	LSTM	RF	HAR	MIDAS
1 year put 5	**0.1421**	0.1436	0.1714	0.1562	0.1544
1 year put 50	**0.0972**	0.0992	0.1124	0.1930	0.1012
1 year call 5	**0.1195**	0.1232	0.1359	0.1246	0.1219
MAPE					
1 week put 5	**0.0685**	0.0702	0.0868	0.0762	0.0731
1 week put 50	**0.0697**	0.0715	0.0886	0.0764	0.0731
1 week call 5	**0.0648**	0.0662	0.0789	0.0691	0.0677
1 month put 5	**0.0359**	0.0357	0.6055	0.0383	0.0376
1 month put 50	**0.0348**	0.0355	0.2640	0.0369	0.0360
1 month call 5	**0.0332**	0.0244	0.0376	0.0344	0.0339
3 months put 5	**0.0247**	0.0244	0.2346	0.0264	0.0262
3 months put 50	**0.0233**	0.0233	0.1757	0.0248	0.0243
3 months call 5	**0.0225**	0.0120	0.2000	0.0233	0.0231
6 months put 5	**0.0193**	0.0192	0.1963	0.0209	0.0205
6 months put 50	**0.0177**	0.0200	0.1445	0.0191	0.0186
6 months call 5	**0.0172**	0.0156	0.1649	0.0179	0.0176
1 year put 5	**0.0153**	0.0144	0.1715	0.0168	0.0164
1 year put 50	**0.0139**	0.0140	0.1124	0.0152	0.0145
1 year call 5	**0.0134**	0.0382	0.1360	0.0140	0.0136

First column indicates the level of moneyness measured in delta for the different maturities. Delta 50 is the ATM option, and delta 5 is the OTM put and call option. The highlighted value indicates the best-fitted value for that particular option for MSE, RMSE, MAE and MAPE, respectively

5.2 One week to maturity

For all other options with a maturity of one week, the LSTM model outperforms the benchmark models on RMSE and MAE. Exceptions are a put option with a delta of 35, where the benchmark GARCH model has an MAE 1.96% lower than the LSTM, and a put option with a delta of 10, for which both models have an RMSE of 0.7635, a forecast accuracy which is 7.85% better than the random forest model. On average, for all options with one week to maturity, the LSTM outperforms the GARCH model with 0.75% in RMSE and 2.77% measured by MAE. Also, the LSTM performs 11.07% and 14.99% better than the random forest in terms of RMSE and MAE, respectively. These findings show that the benchmark AR-GARCH model is not significantly poorer than the LSTM model at shorter maturities, while the LSTM clearly outperforms the random forest model. For the one-week maturity, there is no significant difference between LSTM and RF, according to the DM test.

In Figs. 8, 9 and 10, the forecasted values for the ATM put option with one week to maturity are plotted against the actual spot of implied volatility for the forecasting period for LSTM, RF and benchmark GARCH, respectively.

5.3 One month to maturity

When time to maturity increases to one month, the results fluctuate more. For the OTM and ATM put and call options depicted in Table 5, the benchmark AR-GARCH model seems to deliver the best forecasting accuracy of the three models measured by RMSE. The LSTM performs equally well for the OTM call option and beats the benchmark AR-GARCH in terms of MAE for the OTM put option. In terms of RMSE for all levels of moneyness, the benchmark AR-GARCH model performs on average 0.78% better than the LSTM model. However, the LSTM is, on average, 1.56% more accurate measured by MAE. The benchmark AR-GARCH model captures the outliers, i.e., significant sudden changes in volatility, better than the LSTM model. Being well equipped to capture volatility clustering and fat-tailed returns, the AR-GARCH is a tough benchmark to beat, even for the LSTM model. We note that the LSTM model outperforms both benchmark models in terms of RMSE and MAE for OTM call options, i.e., options with a delta of 25 and lower. Further, the RMSE and MAE are 4.97% and 10.82% lower for the LSTM model than for the random forest model. The LSTM model outperforms the random forest model more often for call options than for put options.

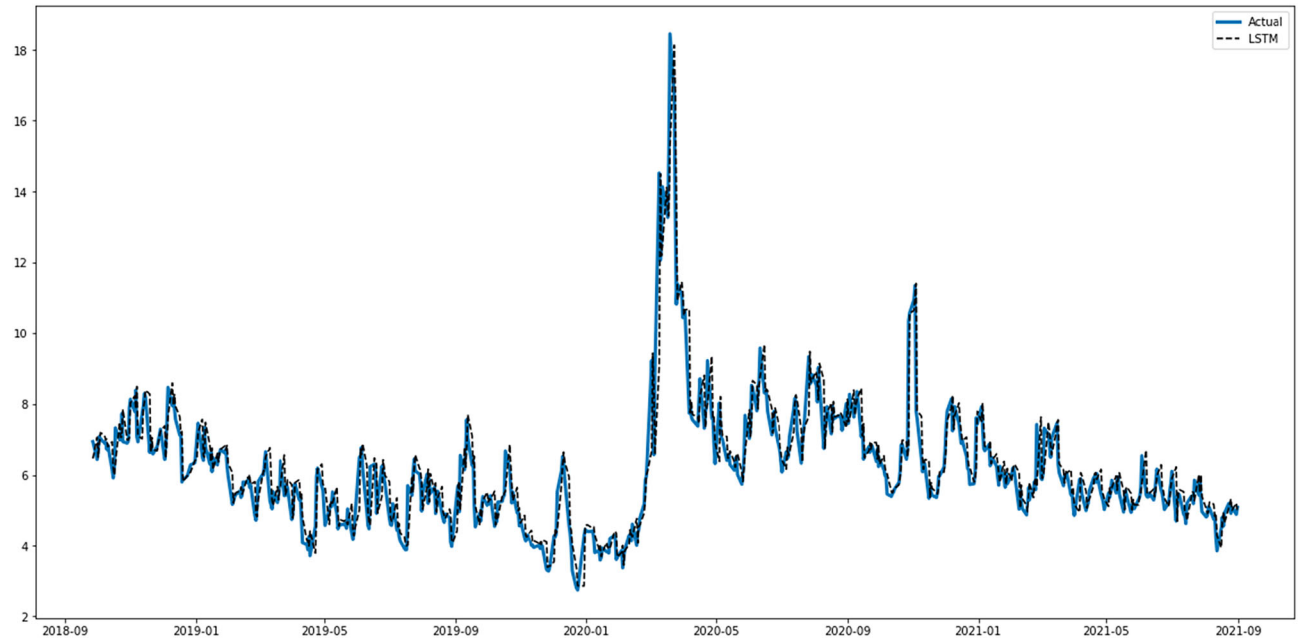

Fig. 8 Forecast results for LSTM model on ATM one-week to maturity option. Forecast results for LSTM model on ATM one-week to maturity option plotted against the actual spot rate for the implied volatility

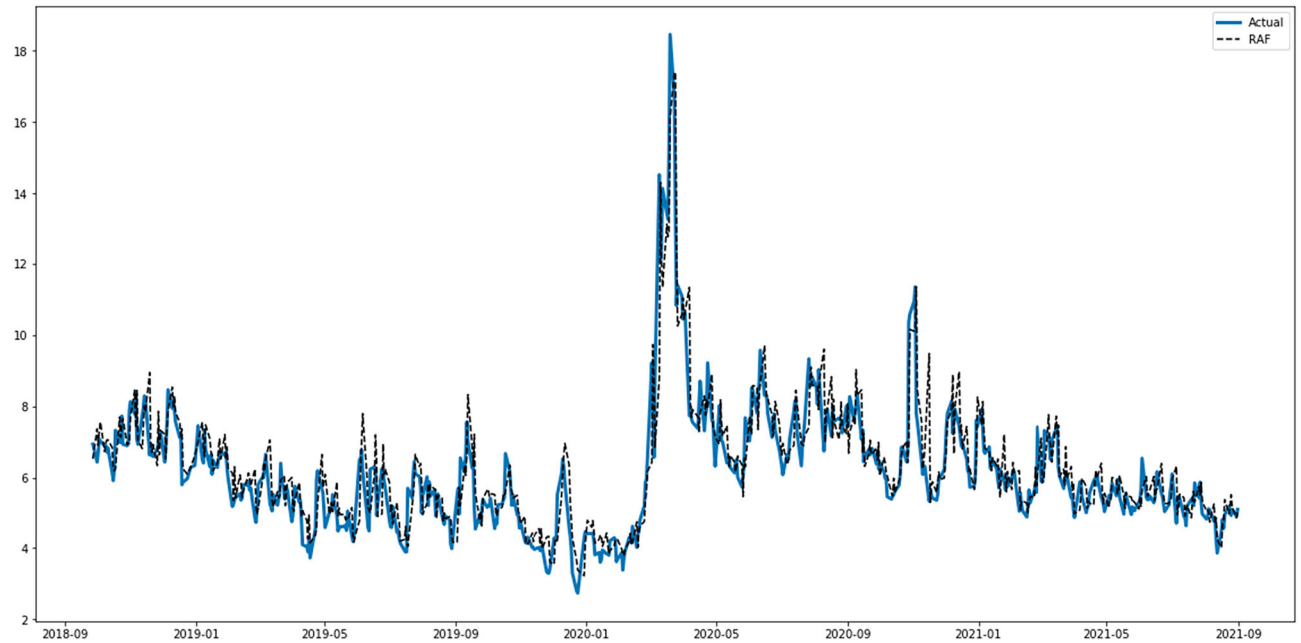

Fig. 9 Forecast results for RF model on ATM one-week to maturity option. Forecast results for random forest model on ATM one-week to maturity option plotted against the actual spot rate for the implied volatility

According to the DM test, LSTM is significantly better than the RF for OTM call options, but not significantly better for ATM and OTM put options. The same results apply comparing LSTM to the AR-GARCH model. LSTM is significantly better for OTM call options, but there is no significant difference for ATM and OTM put options. AR-GARCH delivers significantly poorer forecasting accuracy for ATM and OTM put options than RF, but there are no significant differences in forecasting accuracy between these models for OTM call options.

Figures 11, 12 and 13 plot the forecasted values of implied volatility for ATM put options expiring in one month for the LSTM, RF and AR-GARCH model. The plot shows that the RF model has problems with the extensive shocks in

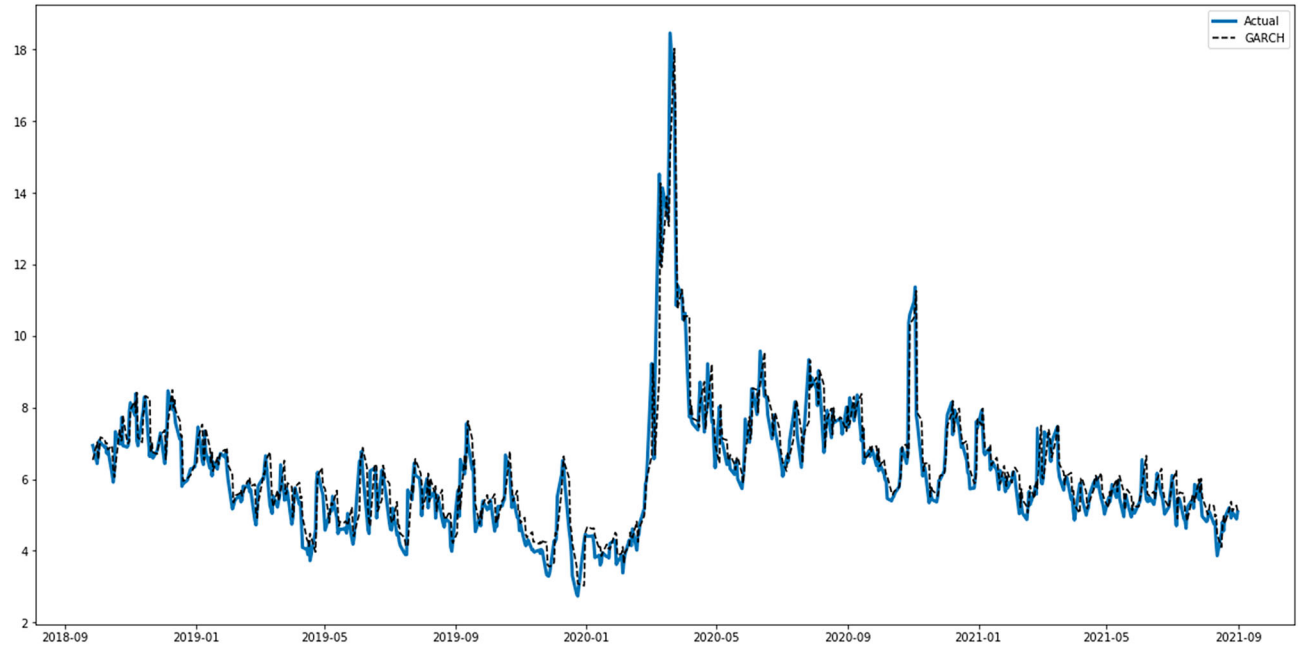

Fig. 10 Forecast results for benchmark AR-GARCH model on ATM one-week to maturity option. Forecast results for the benchmark AR-GARCH model on ATM one-week to maturity option plotted against the actual spot rate for the implied volatility

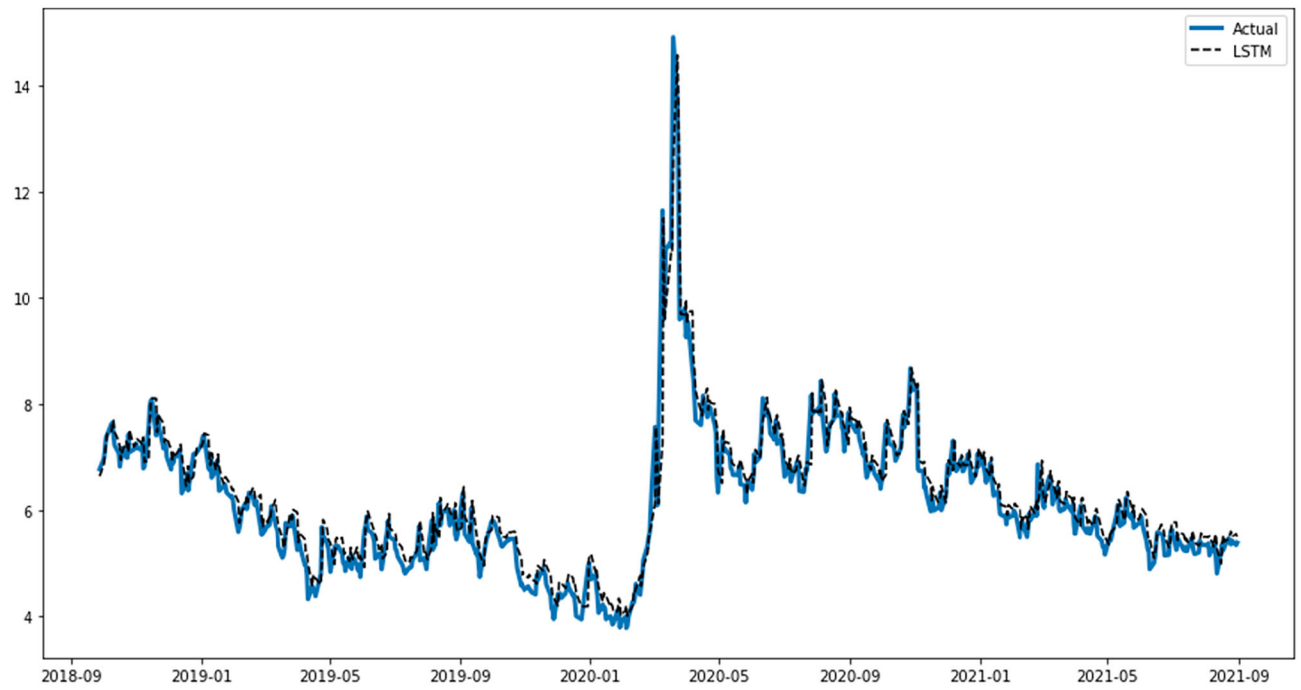

Fig. 11 Forecasting results for LSTM model on ATM one-month option. Forecast results for LSTM model on ATM one-month to maturity option plotted against the actual spot rate for the implied volatility

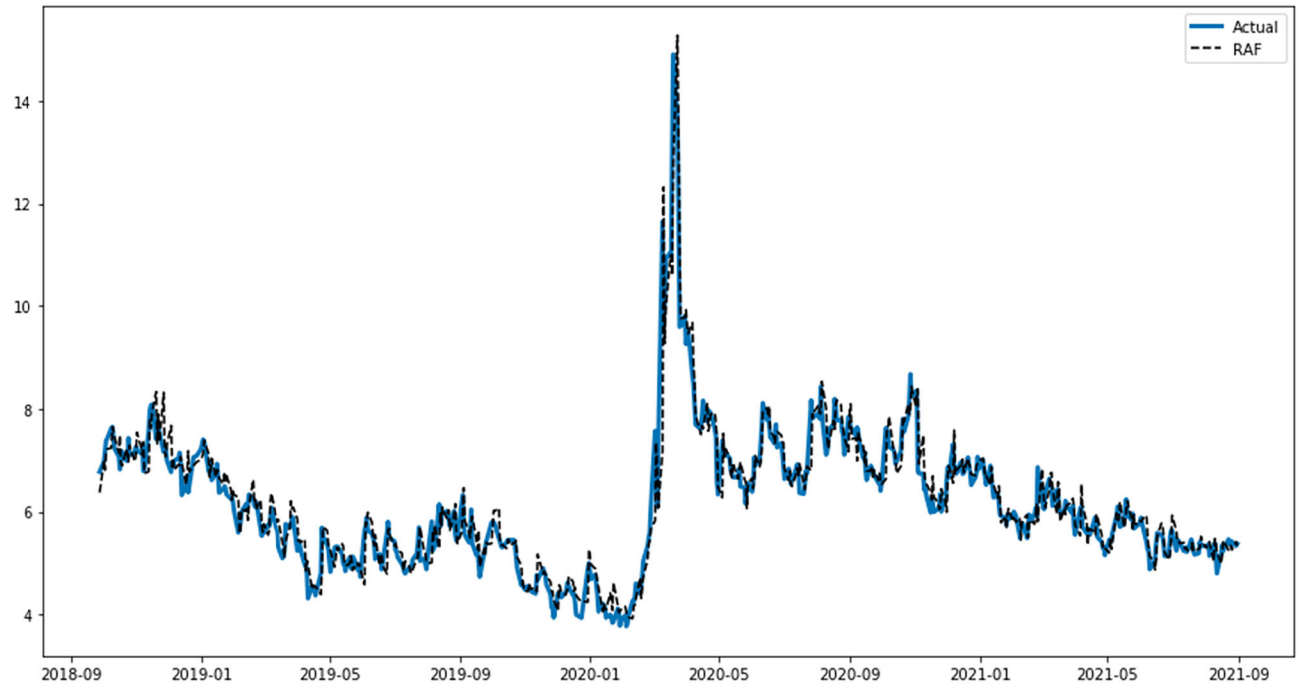

Fig. 12 Forecasting results for random forest model. Forecast results for random forest model on ATM one month to maturity option plotted against the actual spot rate for the implied volatility

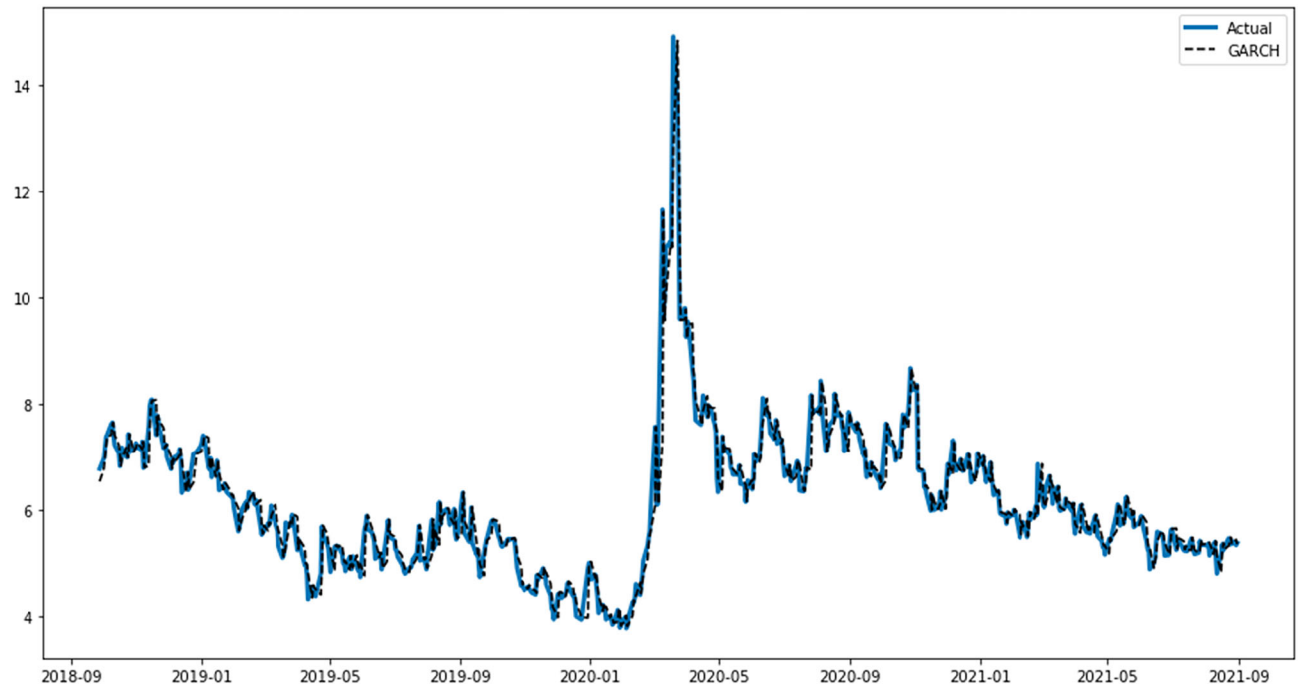

Fig. 13 Forecasting results for benchmark AR-GARCH model. Forecast results for AR-GARCH model on ATM one month to maturity option plotted against the actual spot rate for the implied volatility

implied volatility, especially around March 2020, when the COVID-19 pandemic had its outbreak worldwide. The RF overestimates the peaks from the COVID-19 shocks, whereas the LSTM model underestimates these shocks. All through the test period, which stretches from the end of September 2018 to august 2021, the AR-GARCH fits the rapid changes in implied volatility better than the machine learning models, especially around the extensive shocks, when implied volatility rises significantly.

5.4 Longer maturities

For all options maturing in three months and beyond, the simple benchmark AR(1)-GARCH(1,1) model proved superior to the more complicated machine learning models across all moneyness levels. Due to non-stationarity at a 5% significance level, first difference is applied to the time series of all options maturing beyond three months. The benchmark AR(1)-GARCH(1,1) model performs better in terms of forecasting accuracy than the LSTM model with increasing maturity. At the same time, the Random Forest comes closer to the LSTM with increasing maturity, measured in average RMSE. However, LSTM still outperforms the random forest for all moneyness levels except for a three-month put option with a delta of five and a call option with a delta of 35. On average, the difference in RMSE between the random forest and LSTM declines from 11.07% for one-week options to 2.80% for a one-year option. Measured in MAE, the difference between these models is larger, varying from the lowest for the three-month option at 8.24–14.99% for the one-week option. Interestingly, the MSE increases between the two models from 10.08% for the option maturing in six months to 11.25% for the option maturing in one year, while the difference in RMSE decreases. The DM test for the longer maturities indicates no statistically significant difference between the forecasts. This result is expected as the day-to-day changes in implied volatility decrease as maturity increases.

5.5 Other findings

The distribution for the changes in implied volatility has high peaks, fat tails and changes with time to maturity. Regressing the benchmark AR(1)-GARCH(1,1) model in sample, we assume that the residuals follow a normal distribution. Performing the same regressions assuming Student's t-distributed errors, the average RMSE declined by 1.36% for one-week to maturity options. For options maturing in one month, the Student's t-distributed model performs 0.24% better than the benchmark AR-GARCH model with normally distributed errors. Apparently, the Student's t-distribution fits data with mean clustering and fat tails better than the normal distribution. Our findings further indicate that the t-distributed models fit the data better for *shorter* maturities. The benchmark AR(1)-GARCH(1,1) model with normally distributed errors is better than the t-distribution for the first-order integrated options maturing in three months and beyond. We further note that the in-sample goodness of fit decreases with maturity for the Student's t-distributed AR(1)-GARCH(1,1) model compared to the normally distributed AR(1)-GARCH(1,1) benchmark model.

In empirical forecasting applications the choice of forecast evaluation metrics is an important consideration. As extensively discussed by Hewamalage et al. [19], it is not always clear what the appropriate evaluation metric is, as this depends on the application. To accommodate this, we compute both MSE, RMSE, MAE and MAPE, as presented in Sect. 5. Overall, the main results are broadly consistent across these metrics. In this study, we believe MSE is the most appropriate evaluation metric since the machine learning models are trained using this loss function. We note that scaling absolute forecast errors, as expressed by MAPE, slightly shifts model preferences in favor of LSTM, while retaining the main result that the AR-GARCH model is fully up to par with this more complex machine learning model.

6 Conclusions and future research

The main objective of this study is to compare the predictive power of LSTM models with that of Random Forest and well-established time series models, for forecasting implied volatility of currency options. All regressions are conducted on daily observations of the spot rate of implied volatility for EUR/USD FX options. Volatility forecasts are of interest to market participants for hedging and trading purposes. Also, central banks need to have a view on exchange rates, because changes in exchange rates impact inflation in open economies. This is particularly true for small open economies. To our knowledge, there are only a few prior studies applying machine learning models to forecasting the *implied volatility* of currency options (ref. Sect. 2 above).

We find that the AR-GARCH model outperforms the LSTM model for longer maturities, and that the RF model is the poorest overall forecaster. LSTM is the better model for shorter maturities. Shorter maturity options are more volatile than options with longer maturities. The LSTM seems to capture rapid changes in implied volatility better than the benchmark models, which is consistent with the findings in the existing literature. The LSTM model is able to capture immense and immediate changes in implied volatility, which is important for hedging against significant shifts in FX rates.

Overall, the Random Forest model is a poorer forecaster of implied volatility than the LSTM and AR-GARCH model for all moneyness levels and time to maturity.

Several multivariate empirical models have been explored in the literature, such as VARs and VECMs. An extension to this study could include examining Bayesian VARs, which can be compared with machine learning approaches like BNNs (Bayesian neural network), using implied volatility of options with different maturity profiles and moneyness as explanatory variables. The BNN had the best performance of the ML methods examined in the so called M3 forecasting competition conducted by Makridakis [25].

One might also take a deep learning approach to identifying exogenous variables in prediction models for long-term forecasts, similar to the PCA analysis of Papailias et al. [32]. The idea here is basing the deep learning algorithms on the value-adding conditions of exogenous variables presented by Bojer and Meldgaard [5], Makridakis [25].

Author contributions Olsen and Djupskås implemented models and dataset and wrote draft manuscript. de Lange and Risstad formulated research questions, implemented two additional benchmark models and metrics, reviewed and rewrote the manuscript.

Funding Open access funding provided by NTNU Norwegian University of Science and Technology (incl St. Olavs Hospital - Trondheim University Hospital). No funding was obtained for this manuscript.

Declarations

Competing interests The authors declare no competing interests.

References

1. Babii, A., Ghysels, E., Striaukas, J.: Machine learning time series regressions with an application to nowcasting. J. Bus. Econ. Stat. **40**(3), 1094–1106 (2022)
2. Bharadia, M.A.J., Christofides, N., Salkin, G.R.: A quadratic method for the calculation of implied volatility using the Garman-Kohlhagen model. Financ. Anal. J. **52**(2), 61–64 (1996)
3. Black, F., Scholes, M.: The pricing of options and corporate liabilities. J. Polit. Econ. **81**, 637–659 (1973)
4. Bollerslev, T.: Generalized autoregressive conditional heteroskedasticity. J. Econom. **31**(3), 307–327 (1986). https://doi.org/10.1016/0304-4076(86)90063-1
5. Bojer, C.S., Meldgaard, J.P.: Kaggle Forecasting Competitions: An Overlooked Earning Opportunity. International Journal of Forecasting (2020) URL: https://www.sciencedirect.com/science/article/pii/S0169207020301114
6. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001). https://doi.org/10.1023/A:1010933404324
7. Brownlee, J.: Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras (2016). https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/
8. Carr, P., Wu, L., Zhang, Z.: Using machine learning to predict realized variance. J. Investment Manag. **18**(2), 1–16 (2020). https://doi.org/10.48550/arXiv.1909.10035
9. Chollet, F. et al.: Keras. GitHub (2015). https://github.com/fchollet/keras
10. Corsi, F.: A simple approximate long-memory model of realized volatility. J. Financ. Economet. **7**(2), 174–196 (2009)
11. Enders, W.: Applied Econometrics Time Series, 4th edn. Wiley, New York (2015)
12. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**, 3–42 (2006). https://doi.org/10.1007/s10994-006-6226-1
13. Ghysels, E., Santa-Clara, P., Valkanov, R.: Predicting volatility: getting the most out of return data sampled at different frequencies. J. Econom. **131**, 59–95 (2006)
14. Ghysels, E., Marcellino, M.: Applied Economic Forecasting Using Time Series Methods. Oxford University Press (2018)
15. Glosten, L.R., Jagannathan, R., Runkle, D.E.: On the relation between the expected value and the volatility of the nominal excess return on stocks. J. Financ. **48**, 1779–2180 (1993). https://doi.org/10.1111/j.1540-6261.1993.tb05128.x
16. Hosker, J.J., Djurdjevic, S., Nguyen, H., Slater, R.D.: Improving VIX futures forecasts using machine learning methods. SMU Data Sci. Rev. **1**, 4 (2018)
17. Haug, E.G., Frydenberg, S., Westgaard, S.: Distribution and statistical behavior of implied volatilities. Bus. Valuation Rev. **29**(4), 186–199 (2010). https://doi.org/10.5791/0897-1781-29.4.186
18. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, 2nd edn. Springer (2014)
19. Hewamalage, H., Ackermann, K., Bergmeir, C.: Forecast evaluation for data scientists: common pitfalls and best practices. Data Min. Knowl. Disc. **37**(2), 788–832 (2023)
20. James, G., Witten, D., Hastie, T.: An Introduction to Statistical Learning: With Applications in R, 2nd edn. Springer (2021)
21. Kandel, I., Castelli, M.: The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. ICT Express **6**(4), 312–315 (2020). https://doi.org/10.1016/j.icte.2020.04.010
22. Kryzanowski, L., Galler, M.: Using artificial neural networks to pick stocks. Financ. Anal. J. **49**(4), 21–27 (1993)
23. Krauss, C., Xuan, M., Huck, N.: Deep neural networks, gradient boosted trees, random forests: statistical arbitrage on the S&P 500. Eur. J. Oper. Res. **259**(2), 689–702 (2017). https://doi.org/10.1016/j.ejor.2016.10.031
24. Lim, C.M., Sek, S.K.: Comparing the performances of GARCH-type models in capturing the stock market volatility in Malaysia. Procedia Econ. Finance **5**, 478–487 (2013)
25. Makridakis, S.: Statistical, machine learning and deep learning forecasting methods: comparisons and ways forward. J. Oper. Res. Soc. (2022). https://doi.org/10.1080/01605682.2022.2118629
26. Mandelbrot, B.: The variation of certain speculative prices. J. Bus. **36**(4), 394–419 (1963)
27. McDonald, R.L.: Derivatives Markets, 3rd edn. Pearson Education Limited, London (2014)
28. Medvedev, N., Wang, Z.: Multistep forecast of the implied volatility surface using deep learning. J. Futur. Mark. **42**(4), 645–667 (2022)

29. Namin, A., Namini, S.: Forecasting economics and financial time series: arima vs. LSTM (2018). https://doi.org/10.48550/arXiv.1803.06386
30. Ni, H., Dong, X., Zheng, J., Yu, G.: An Introduction to Machine Learning in Quantitative Finance. World Scientific, Chapter 5.4. Print ISSN: 2059–769X, Online ISSN: 2059–7703 (2021)
31. Ornelas, J.R.H., Mauad, R.B.: Implied volatility term structure and exchange rate predictability. Int. J. Forecast. **35**, 1800–1813 (2019)
32. Papailias, F., Thomakos, D.D., Liu, J.: The baltic dry index: cyclicalitites, forecasting and hedging strategies. Empir. Econ. (2016). https://doi.org/10.1007/s00181-016-1081-9
33. Pedregosa, et al.: Sckit-learn: Machine Learning in Python. JMLR **12**, 2825–2830 (2011)
34. Poon, S., Granger, C.: Forecasting financial market volatility: a review. Dep. Econom. **1**, 1 (2001). https://doi.org/10.2139/ssrn.268866
35. Ramasamy, R., Munisamy, S.: Predictive accuracy of GARCH, GJR and EGARCH models select exchange rates application. Glob. J. Manag. Bus. Res. **12**(15), 89–100 (2012)
36. Schmidt, L.: Volatility Forecasting Performance of GARCH Models: A Study on Nordic Indices During COVID-19. [Master thesis] Umeå University (2021)
37. Sjölander, P.: A stationary unbiased finite sample ARCH-LM test procedure. Appl. Econom. **43**(8), 1019 (2010). https://doi.org/10.1080/00036840802600046
38. Vrontos, S., Galakis, J., Vrontos, I.: Implied volatility directional forecasting: a machine learning approach. Quant. Finance **21**(4), 1–20 (2021). https://doi.org/10.1080/14697688.2021.1905869