Quantitative Finance

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/rquf20

On the pricing of capped volatility swaps using machine learning techniques

Stephan Höcht, Wim Schoutens & Eva Verschueren

To cite this article: Stephan Höcht, Wim Schoutens & Eva Verschueren (2024) On the pricing of capped volatility swaps using machine learning techniques, Quantitative Finance, 24:9, 1287-1300, DOI: 10.1080/14697688.2024.2305643

To link to this article: https://doi.org/10.1080/14697688.2024.2305643

Quantitative Finance, 2024
Vol. 24, No. 9, 1287–1300, https://doi.org/10.1080/14697688.2024.2305643

On the pricing of capped volatility swaps using machine learning techniques

STEPHAN HÖCHT†, WIM SCHOUTENS*‡ and EVA VERSCHUEREN‡

†Assenagon Asset Management S.A., München, Germany
‡Department of Mathematics, KU Leuven, Leuven, Belgium

(*Received 13 June 2023; accepted 4 January 2024; published online 6 February 2024*)

A capped volatility swap is a forward contract on an asset's capped, annualized, realized volatility, over a predetermined period of time. This paper presents data-driven machine learning techniques for pricing such capped volatility swaps, using unique data sets comprising both the strike price of contracts at initiation and the daily observed prices of running contracts. Additionally, the developed model can serve as a validation tool for external volatility swap prices, flagging prices that deviate significantly from the estimated value. In order to predict the capped, future, realized volatility, we explore distributional information on the underlying asset, specifically by extracting information from the implied volatilities and market-implied moments of the asset. The pricing performance of tree-based machine learning techniques and a Gaussian process regression model is evaluated in a validation setting tailored to the use of financial data.

Keywords: Capped volatility swaps; Pricing; Implied volatility; Market-implied moments; Gaussian process regression; Tree-based machine learning

1. Introduction

The popularity of volatility derivatives, including volatility swaps, has witnessed significant growth over the years, in line with the enduring prominence of volatility as a risk measure in the financial industry and the growing importance of volatility as an asset class on its own. A volatility swap is a bilateral contract wherein the future realized volatility of the underlying asset, quoted in annual terms, over a predetermined period of time, is traded against a fixed volatility strike. To mitigate the risk exposure of such contracts, the realized volatility is often capped at 2.5 times the strike, resulting in what is called a capped volatility swap.

The payoff of the instrument is given by (Flavell 2010)

$$\text{payoff} = \text{notional} \times [\min(\text{cap},\ \text{realized volatility}) - \text{strike}]. \tag{1}$$

The strike, denoted by K, is fixed at contract initiation as the fair delivery price of the contract. The realized volatility σ_R is determined at contract expiration as

$$\sigma_R = 100 \times \sqrt{\frac{252}{N}\sum_{i=1}^{N}\left(\ln\left[\frac{S_i}{S_{i-1}}\right]\right)^2}, \tag{2}$$

with S_i the closing price of the underlying asset at trading day i and i running over a total of N trading days between contract initiation and expiration at maturity T.

Volatility swaps allow investors to get a pure exposure to the volatility of the underlying asset, making the product an interesting instrument for both hedging and speculative purposes (Demeterfi *et al.* 1999, Carr and Lee 2009). The pricing of capped volatility swaps is inextricably linked to the trading of such contracts. Specifically, pricing is involved in the negotiation of the conditions of a newly initiated contract with the counterparty, i.e. when determining the fair strike, and in continuously monitoring the value of running volatility swaps. As exotic derivatives, these instruments are traded over-the-counter, with no price readily available on exchange. Consequently, market participants often rely on external entities, such as financial market data providers, to price a contract. However, the prices obtained from external sources are the end product of a provider's proprietary internal model of which the specifics are usually not disclosed. Moreover, occasionally, prices from different sources tend to

*Corresponding author. Email: wim.schoutens@kuleuven.be, wim.schoutens@wis.kuleuven.be

deviate substantially, hampering a correct valuation of the product. As such, the present work aims to establish the foundations of a practical pricing tool for capped volatility swaps, to gain insight into the value of a specific contract and to validate the prices from external sources, flagging prices that deviate substantially from the estimated value.

The price of a capped volatility swap, at any point t in time, is determined as

$$\text{Price}_t = \text{DF}_t \times \mathbb{E}_t^{\mathcal{Q}} \left[\min\left(\text{cap}, \sigma_R\right) - K \right], \tag{3}$$

with discount factor DF_t, unit notional and expectations taken under a pricing measure $\mathcal{Q}$ (Ross 1976, Harrison and Pliska 1981). The discount factor $\text{DF}_t = \exp(-r(T-t))$ is determined by the risk-free interest rate r in the market and the maturity T of the derivative product. To build a pricing tool, one could consult the currently existing pricing literature. This branch of literature is highly focused on model-based pricing approaches, taking modeling assumptions on the asset process driving the underlying asset of the product. However, the non-linear nature of the capped volatility swap's payoff structure, as manifested in the cap level and the square root operator, results in a complex pricing expression. Many authors therefore first derive an expression for the price of a related product, the variance swap, as a starting point to find a solution to the pricing problem of (capped) volatility swaps.

The payoff structure of the variance swap is similar to the one described in Equation (1), but based on the realized variance, and without imposing the cap level. We refer to recent contributions of Kim and Kim (2020), Rujivan and Rakwongwan (2021) and Salmon and SenGupta (2021) for an expression of the fair delivery strike of a variance swap under, respectively, the log-normal SABR model, the Black-Scholes model and the BN-S model. Various adaptations of the Heston stochastic volatility model lead to an analytical expression for this strike in Issaka (2020), Lin and He (2020) and He and Zhu (2022). Issaka (2020) is the sole author extending the results to obtain a price for the variance swap at any point t in time.

A transition from variance swaps to uncapped volatility swaps can be pursued by leveraging the result of Brockhaus and Long (2000):

$$\mathbb{E}\left[\sqrt{\sigma_R^2}\right] \approx \sqrt{\mathbb{E}[\sigma_R^2]} - \frac{\text{Var}\left[\sigma_R^2\right]}{8 \cdot \mathbb{E}[\sigma_R^2]^{\frac{3}{2}}}. \tag{4}$$

The approximation in Equation (4) is used in Issaka (2020) and Salmon and SenGupta (2021) to find an approximate, analytical expression for the delivery strike of a volatility swap. Issaka (2020) extends the results to a solution for Equation (3), at any point t in time, but without imposing the cap level. An approximate formula for the strike is directly derived from underlying model assumptions in Lin and He (2020), Kim and Kim (2020), Rujivan and Rakwongwan (2021) and He and Zhu (2022).

The applicability of the currently existing literature in building a practical pricing tool for capped volatility swaps is limited, for multiple reasons. First, a closed-form expression for the fair delivery strike of a volatility swap is very rarely discussed, and often only approximations are derived. Second, the literature is particularly oriented towards finding this fair price of the contract at initiation, with limited extensions to pricing the product at any point t in time. Additionally, the issue of pricing capped volatility swaps has not been addressed, despite the fact that, in practice, the presence of a cap level is market standard. Finally, the absence of a closed-form expression for the price of a capped volatility swap, under the assumptions of an asset pricing model, could be addressed by employing numerical methods, such as (adaptations of) a Monte Carlo simulation (Acworth *et al.* 1998). This approach however involves the use of random sampling and is renowned for its computational intensity and time consumption (De Spiegeleer *et al.* 2018). Moreover, both closed-form expressions and Monte Carlo frameworks for pricing are derived based on assumptions regarding the asset pricing process. These assumptions introduce calibration risk, as illustrated in the work of Guillaume and Schoutens (2012), and do not guarantee alignment with market prices.

We use the aforementioned argumentation to shift focus from these model-based methods to a data-driven pricing approach, deploying machine learning techniques. Lately, machine learning has already proven to be successful in solving challenges in finance. We refer to De Spiegeleer *et al.* (2018), Gan *et al.* (2020) and Davis *et al.* (2020) for recent applications on the pricing of financial products. To the best of our knowledge, we are the first to develop a data-driven machine learning solution in the context of pricing capped volatility swaps. Using verified market data on the contracts, we are able to predict prices, not only at contract initiation but also during the lifetime of the product. A data-driven price reflects the market standard as it historically has been observed. Any substantial discrepancy between the predicted price and an external price can be flagged and investigated.

The paper is organized as follows. Section 2 presents the data sets and introduces the input variables used in the pricing of capped volatility swaps. We make a distinction between two types of data sets, one focused on time series of prices of running volatility swaps and one solely focused on delivery strike prices at contract initiation. Section 3 theoretically outlines the examined machine learning algorithms as well as the tailored validation setting that is used to tune and test these algorithms. The pricing performance of tree-based machine learning methods and a Gaussian process regression model is discussed in Section 4. Section 5 concludes.

2. Market data on capped volatility swaps

The data sets† categorized as type I consist of time series of prices of multiple capped volatility swap contracts, written on the same underlying asset. A general overview is provided in Table 1. The price for each individual contract is recorded on a daily basis, spanning the period between the contract's initiation and settlement dates. Given the temporal overlap of some contracts, multiple observations are captured on the same trading day. Our study encompasses 3 different data sets

† Both data sets (type I and type II) stem from an internal database comprised of traded volatility swaps and contain contractual information for each volatility swap.

Table 1. Overview of the available data sets of type I, on running capped volatility swap contracts.

	S&P 500	AAPL	JPM
Time span	Nov. 2018–Dec. 2022	Nov. 2018–Sept. 2022	Nov. 2018–Sept. 2022
Number of contracts	462	228	193
Number of observations	81 153	35 496	26 990

Table 2. Overview of the data sets of type II on capped volatility swaps at contract initiation.

	Index	Equity
Time span	Jan. 2017–Nov. 2022	Jan. 2017–Nov. 2022
Number of observations	826	8197
Number of underliers	8	235

of type I, with contracts written on the S&P 500 stock index as well as on individual stocks of Apple Inc. (AAPL) and JPMorgan Chase & Co (JPM). The data sets of type I will be used to construct a model for pricing capped volatility swaps at any given time t, over the course of the contract.

The data sets of type II consist of fair delivery prices of capped volatility swaps at contract initiation. These data sets will solely serve the purpose of constructing a pricing model at $t = 0$. A general overview of these data sets is provided in Table 2. We consider one set of observations written on stock indices and another set written on individual stocks. A remarkable difference with the data sets of type I is the inclusion of contracts written on various underliers, albeit of the same asset type, within a single data set.

2.1. The response variable

When analyzing Equation (3), we observe that various factors, including the strike K and remaining contract duration, impact the price (PX) of a capped volatility swap, both at initiation and over the course of the contract. While most factors are fixed at a certain point t in time, estimating the price of a contract requires a prediction for the total realized volatility σ_R. Therefore, we use a revised pricing equation, based on Equation (3), that extracts the already realized part, the accrued volatility (AcVOL), at time t, from the total realized volatility σ_R:

$$\begin{aligned} \mathtt{PX}_t &= \mathrm{DF}_t \times \left(\mathbb{E}_t^{\mathcal{Q}}\left[\min\left(\mathrm{cap}, \sigma_R\right)\right] - K\right); \\ &= \mathrm{DF}_t \times \left(\sqrt{\begin{matrix}\mathtt{IVOL}_t^2 \times (1 - \mathtt{Weight}_t) \\ +\mathtt{AcVOL}_t^2 \times \mathtt{Weight}_t\end{matrix}} - K\right). \end{aligned} \tag{5}$$

We hereby introduce the variables Weight, AcVOL and IVOL. Weight reflects the proportion of contract duration that has elapsed since the initiation, until time t. AcVOL is calculated as

$$\mathtt{AcVOL}_t = 100 \times \sqrt{\frac{252}{n}\sum_{i=1}^{n}\left(\ln\left[\frac{S_i}{S_{i-1}}\right]\right)^2}, \tag{6}$$

with n the number of trading days between the initiation of the contract and time t. Note that, at time t, both $\mathtt{Weight}_t$ and $\mathtt{AcVOL}_t$ are known variables. Finally, IVOL is determined to match the equality in Equation (5) and comprises the unrealized part of the total volatility, the expected value and the cap level, i.e. it isolates the information that necessitates prediction, in order to come up with a value for $\mathtt{PX}_t$.

Data type I. Instead of focusing directly on the price of a contract, we focus on modeling IVOL, a forward-looking value, reflecting the expected volatility over the remaining duration of the contract, taking into account the cap level set on the total realized volatility. The relation in Equation (5) can be used to derive an estimate price, based on a prediction for IVOL.

Figure 1(a–c) present the distribution and time-ordered values of IVOL across the data sets of type I. We observe a slightly right-skewed distribution, defined over the positive real line, with a shift to higher IVOL values for contracts written on individual stocks. The observed jump in March 2020 exactly coincides with the beginning of the COVID-19 period. We note that, in both the S&P 500 and JPM data sets, certain observations for IVOL exhibit a decline to zero, which occurs when the respective contract approaches the cap level.

Data type II. Given that the strike K is determined to ensure the fairness of the contract at initiation, it can be inferred from (5), with $t = 0$, that $K = \mathtt{IVOL}_0$. As a result, the primary focus is on the modeling of the contract's strike price, i.e. the variable Strike. Figure 2 respectively displays the distribution and time-ordered values of Strike across the data sets of type II. We observe again a slightly right-skewed distribution, with a clear shift to higher Strike values for contracts written on individual stocks. The data also shows the noticeable impact of the COVID-19 crisis.

2.2. The input variables

A summary of contract specific input features is provided in Table 3. Besides the variables Weight and AcVOL, introduced in Equation (5), we consider the duration and the strike of the contract, stored, respectively, in the variables ITM and Strike. Note that for the data sets of type II, only ITM is a relevant feature. In addition, we examine distributional characteristics of the underlying asset of a contract to estimate IVOL (Type I) and Strike (Type II). This decision is justified by the observation that the value of both variables is essentially determined by (expected) fluctuations in the underlying asset.

2.2.1. Market-implied moments. Since IVOL and Strike are forward-looking parameters, we investigate the market-implied moments, i.e. the moments deduced from the

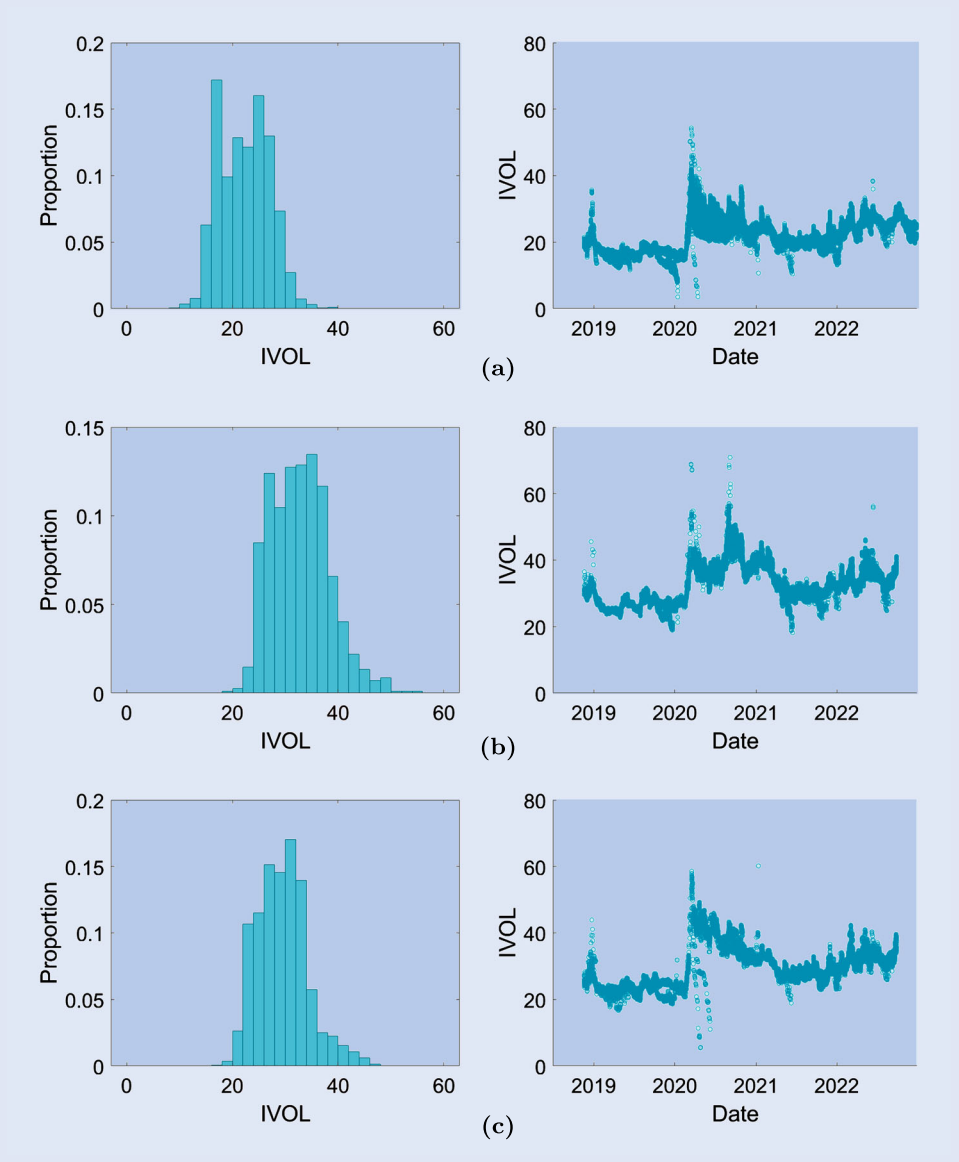

Figure 1. Distribution and time-ordered values of IVOL in data type I, for contracts written on (a) S&P 500, (b) AAPL and (c) JPM.

forward-looking pricing-return density of an asset via quoted vanilla (call and put) options. The theoretical basis for a model-free calculation of these moments is established in Carr and Madan (2001) and Bakshi *et al.* (2003). We use the work of Madan and Schoutens (2016) to state the closed-form expression of the mth moment of an asset's return over a period of length T:

$$\begin{aligned}
&\mathbb{E}\left[\left(\ln\left(\frac{S_T}{S_0}\right)\right)^m\right] \\
&\quad = \left(\ln\left(\frac{\kappa}{S_0}\right)\right)^m + m\left(\ln\left(\frac{\kappa}{S_0}\right)\right)^{m-1} \\
&\qquad \times \left(\frac{F_0}{\kappa} - 1\right) + \exp(rT)\int_0^\kappa \frac{m}{K^2} \\
&\qquad \times \left((m-1)\left(\ln\left(\frac{K}{S_0}\right)\right)^{m-2} - \left(\ln\left(\frac{\kappa}{S_0}\right)\right)^{m-1}\right) \\
&\qquad \times EP(K,T)\,\mathrm{d}K + \exp(rT)\int_\kappa^\infty \frac{m}{K^2} \\
&\qquad \times \left((m-1)\left(\ln\left(\frac{K}{S_0}\right)\right)^{m-2} - \left(\ln\left(\frac{\kappa}{S_0}\right)\right)^{m-1}\right) \\
&\qquad \times EC(K,T)\,\mathrm{d}K, \qquad (7)
\end{aligned}$$

with interest rate r, forward price F_0, a strike κ and expectations taken under a pricing measure. $EC(K,T)$ and $EP(K,T)$ respectively denote the price of a European call and put option at strike K and maturity T. In practice, the integration of traded option prices over a continuum of strikes is avoided by approximating the integral, for instance based on a trapezoidal scheme. Further details on the practical implementation can be found in Madan and Schoutens (2016).

From the raw market-implied moments of the asset's return we infer the market-implied variance, volatility, skewness and kurtosis, introducing $R_T = \ln(\frac{S_T}{S_0})$, and using the following equalities:

$$\mathrm{Var}\,(R_T) = \mathbb{E}\left[R_T^2\right] - (\mathbb{E}\,[R_T])^2\,; \qquad (8)$$

$$\mathrm{Vol}\,(R_T) = \sqrt{\mathrm{Var}\,(R_T)}; \qquad (9)$$

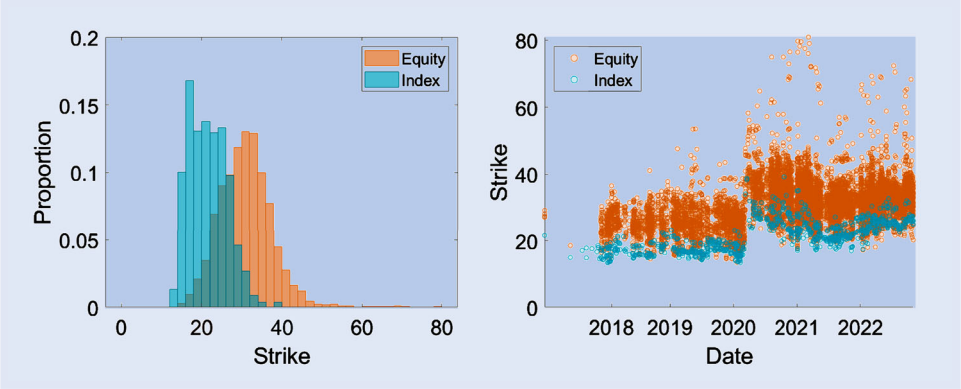

Figure 2. Distribution and time-ordered values of `Strike` in the data sets of type II.

Table 3. Description of contract specific input features.

Variable	Description
`ITM`	The initial time till maturity (duration) of the contract.
`Strike`	The strike of the contract.
`AcVOL`	The realized volatility between initiation and observation time.
`Weight`	The proportion of duration elapsed since initiation until observation time.

$$\text{Skew}\,(R_T) = \frac{\mathbb{E}\left[R_T^3\right] - 3\mathbb{E}\left[R_T\right]\mathbb{E}\left[R_T^2\right] + 2\left(\mathbb{E}\left[R_T\right]\right)^3}{\text{Var}\,(R_T)^{3/2}}; \quad (10)$$

$$\text{Kurt}\,(R_T) = \frac{\mathbb{E}\left[R_T^4\right] - 4\mathbb{E}\left[R_T\right]\mathbb{E}\left[R_T^3\right] + 6\left(\mathbb{E}\left[R_T\right]\right)^2\mathbb{E}\left[R_T^2\right] - 3\left(\mathbb{E}\left[R_T\right]\right)^4}{\text{Var}\,(R_T)^2}. \quad (11)$$

We employ the summary statistics in Equations (9) to (11) as input features to estimate `IVOL` and `Strike`. We hereby explicitly provide information on higher-order moments, incorporating skewness and kurtosis. These statistics serve as indicators for respectively the asymmetry and the tail behavior of the underlying asset's distribution, which is associated with the occurrence of extreme values, and therefore relevant for the cap level imposed on the total realized volatility. The maturity of the market-implied volatility and skewness is matched with the remaining time till maturity of the contract. We fix a low maturity of 30 days for the market-implied kurtosis, to capture the short-term potential on extreme movements in the underlying asset.

Figure 3(a–c) present the 5-day evolution of the market-implied features on the S&P 500 index, over different maturity levels. The minor, temporal fluctuations in market-implied volatility are anticipated due to day-to-day variations in option prices. However, the erratic behavior observed in the skewness, and particularly in the kurtosis, is disproportionate and can be attributed to the approximation required in the estimation of the expression outlined in Equation (7). Moreover, it is anticipated that fluctuations will be particularly prominent for underlying assets with lower liquidity, as a result of restricted access to listed options and the volatile nature of their prices.

2.2.2. Implied volatility. To avoid the necessity of incorporating erratic input features on market-implied measures into our models, we examine an alternative source for features: implied volatilities. Using the Black-Scholes option pricing

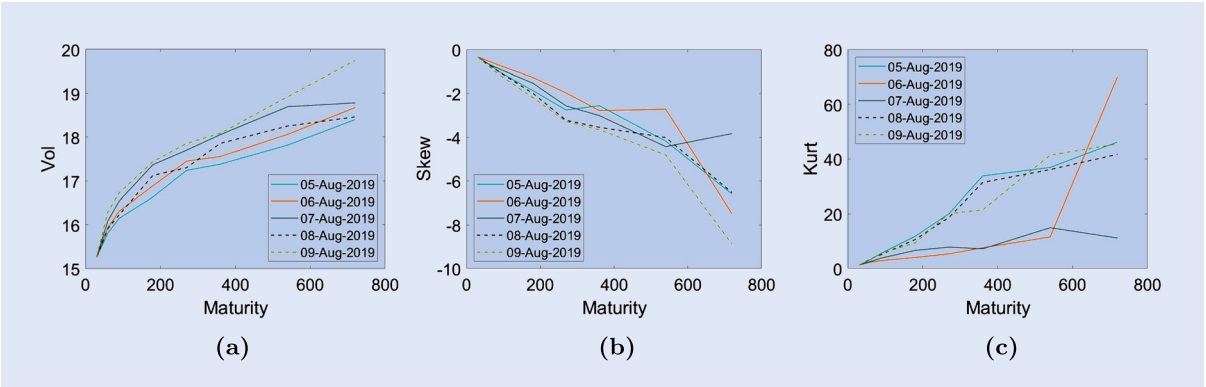

Figure 3. The market-implied (a) volatility, (b) skewness and (c) kurtosis of the S&P 500 over different maturity levels. The evolution is presented in the week of August 5–9, 2019, but similar patterns are observed across the entire sample period.

model, we identify the only free parameter, the volatility level, such that the theoretical model price aligns with the empirical market price of the option. The inferred value is strike and maturity dependent and referred to as the implied volatility, a forward-looking parameter reflecting the market's sentiment on the behavior of the underlying asset. For a discussion on how to estimate implied volatilities in practice, we refer to Bloomberg (2008) and Madan and Schoutens (2016).

An alternative implied volatility-based feature is constructed for each of three market-implied statistics, as introduced in Equations (9) to (11). First, the market-implied volatility of the pricing-return density of the underlying asset is challenged by the implied volatility at the 100% moneyness level. This choice is substantiated through the correlations in the first column of Table 4, for the S&P 500 index. Calculations cover the time span of the S&P 500 data set. We match the maturity of the constructed feature with the remaining time till maturity of the contract.

Following Mixon (2011), the difference between the 75% and 125% implied volatility level, weighted against the 100% implied volatility is constructed as an alternative measure of skewness. Based on the results presented in Table 4, a strong correlation is observed between this alternative measure of skewness and the market-implied skewness, particularly for short-term maturities. It should be noted that the correlation is weaker for a longer duration of 1 and 2 years, which may be attributed to the presence of increased fluctuations in the market-implied skewness for higher maturities. The maturity of the constructed feature is matched with the remaining time till maturity of the contract.

The finance literature has not provided a clear lead in defining a best practice version for an alternative measure of kurtosis, derived from implied volatilities. However, since a measure of skewness is based on the overall symmetry of the implied volatility curve, we propose to examine the tails of the curve in order to develop an own measure of kurtosis. More specifically, we construct a left and right tail measure, using respectively $(\frac{90-95}{100})$% IV and $(\frac{105-110}{100})$% IV. For these measures of kurtosis, we fix the lowest possible maturity of 30 days. Table 4 reveals significant correlation, at the short maturity, between the market-implied kurtosis and alternative measures.

2.2.3. General overview. Table 5 provides an overview of the different feature sets tested for the modeling of `IVOL` and `Strike`. As a baseline configuration, we consider the input settings with only a volatility-based feature derived from market-implied moments (setting S1) and implied volatilities (setting S4). The baseline settings are incrementally extended with higher-order features, meaning that the added value of having a skewness-based feature is tested in setting S2 (based on market-implied moments) and setting S5 (based on implied volatilities). In settings S3 and S6, we additionally add information on short-term kurtosis.

Table 5. Overview of different feature sets for modeling `IVOL` (Type I) and `Strike` (Type II).

	S1	S2	S3	S4	S5	S6
Contract Specific Features	✓	✓	✓	✓	✓	✓
`MI Vol`	✓	✓	✓			
`MI Skew`		✓	✓			
`MI Kurt`			✓			
`100% IV`				✓	✓	✓
$(\frac{75-125}{100})$`% IV`					✓	✓
$(\frac{90-95}{100})$`% IV` & $(\frac{105-110}{100})$`% IV`						✓

3. The modeling setting

We discuss the different candidate models for predictions in the setting of capped volatility swaps: Gaussian process regression and tree-based machine learning techniques. The last part is devoted to the validation scheme that is used to tune the parameters and to test the performance of the models.

3.1. *Gaussian process regression*

Let $(\boldsymbol{X}, \boldsymbol{y}) = \{(\boldsymbol{x}_i, y_i) \mid i = 1, \ldots, n\}$ be a set of observations, with $\boldsymbol{x}_i$ an input vector of dimension p and y_i the corresponding output value. Following Rasmussen and Williams (2006), under the assumptions of the Gaussian process regression (GPR) model and for each $i = 1, \ldots, n$,

$$y_i = f(\boldsymbol{x}_i) + \varepsilon_i, \tag{12}$$

with Gaussian process f and i.i.d. noise $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$.

A Gaussian process is a collection of random variables of which any finite subset has a joint Gaussian distribution. The process is fully characterized by a mean function m, often defined as the zero function, and covariance or kernel function

Table 4. Correlation between the market-implied (MI) features and the implied volatility (IV)-based alternatives for the S&P 500 index, over different maturities.

	`MI Vol`	`MI Skew`	`MI Kurt`	
	`100% IV`	$(\frac{75-125}{100})$`% IV`	$(\frac{90-95}{100})$ `% IV`	$(\frac{105-110}{100})$ `% IV`
30 days	0.9904	−0.8268	0.7956	−0.7100
90 days	0.9866	−0.7144	0.6046	−0.4309
360 days	0.9701	0.0789	−0.1657	−0.2968
720 days	0.9769	0.1985	−0.0195	−0.0423

k. At a given input location $\boldsymbol{x}_i, f(\boldsymbol{x}_i)$ is a random variable that represents a possible output of the true function at this location. The actual observation y_i at input $\boldsymbol{x}_i$ is a sample extracted from the distribution of $f(\boldsymbol{x}_i)$. In general, for a larger sample $(\boldsymbol{X}, f(\boldsymbol{X})) = \{(\boldsymbol{x}_i, f(\boldsymbol{x}_i)) \mid i = 1, \ldots, n\}$ generated from the Gaussian process, it holds

$$f(\boldsymbol{X}) \sim \mathcal{N}\left(m(\boldsymbol{X}) = \boldsymbol{0}, K(\boldsymbol{X}, \boldsymbol{X}) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \ldots & k(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_n, \boldsymbol{x}_1) & \ldots & k(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix}\right), \tag{13}$$

with Gaussian distribution $\mathcal{N}(\cdot, \cdot)$.

For a new set of input values $\boldsymbol{X}^*$, of dimension $n^* \times p$, and unknown corresponding function values $\boldsymbol{f}^*$, the assumptions of a Gaussian process model result in

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}^* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \mathbb{I} & K(\boldsymbol{X}, \boldsymbol{X}^*) \\ K(\boldsymbol{X}^*, \boldsymbol{X}) & K(\boldsymbol{X}^*, \boldsymbol{X}^*) \end{bmatrix}\right), \tag{14}$$

with $\mathbb{I}$ the identity matrix of size n.

The conditional distribution of the unknown $\boldsymbol{f}^*$, given $\boldsymbol{X}, \boldsymbol{y}$ and $\boldsymbol{X}^*$ is then again a multivariate Gaussian defined as

$$\begin{aligned} \boldsymbol{f}^* \mid \boldsymbol{X}^*, \boldsymbol{X}, \boldsymbol{y} \sim \mathcal{N}\Big(& K(\boldsymbol{X}^*, \boldsymbol{X}) \left[K(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \mathbb{I}\right]^{-1} \boldsymbol{y}, \\ & K(\boldsymbol{X}^*, \boldsymbol{X}^*) - K(\boldsymbol{X}^*, \boldsymbol{X}) \\ & \times \left[K(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \mathbb{I}\right]^{-1} K(\boldsymbol{X}, \boldsymbol{X}^*)\Big). \end{aligned} \tag{15}$$

The mean function of the distribution in Equation (15) provides predictions for the values $\boldsymbol{f}^*$. This mean function relies on the choice of covariance function k, which characterizes the covariance between any two $f(\boldsymbol{x})$ and $f(\boldsymbol{x}^*)$. A widely used candidate for the covariance function is the squared exponential kernel $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$, given by

$$k(\boldsymbol{x}, \boldsymbol{x}^*) = \sigma^2 \exp\left(-\frac{|\boldsymbol{x} - \boldsymbol{x}^*|^2}{2l^2}\right), \tag{16}$$

with parameter set $\boldsymbol{\theta} = (\sigma^2, l)$, respectively the signal variance and length-scale parameter. As an alternative to the kernel in Equation (16), one may consider the Matern kernel, which is a generalization of the squared exponential kernel,

$$k(\boldsymbol{x}, \boldsymbol{x}^*) = \frac{1}{\Gamma(\nu) 2^{\nu - 1}} \left(\frac{\sqrt{2\nu}}{l} |\boldsymbol{x} - \boldsymbol{x}^*|\right)^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu}}{l} |\boldsymbol{x} - \boldsymbol{x}^*|\right), \tag{17}$$

with K_{ν} a modified Bessel function.

The above kernel functions are often extended by including an individual length-scale parameter for each of p input features. As an example, the squared exponential kernel is then given by

$$k(\boldsymbol{x}, \boldsymbol{x}^*) = \sigma^2 \exp\left(-\frac{1}{2} \sum_{j=1}^{p} \frac{(x_j - x_j^*)^2}{l_j^2}\right), \tag{18}$$

and $\boldsymbol{\theta} = (\sigma^2, l_1, \ldots, l_p)$.

A common approach to obtain an estimate for the kernel's parameter set $\boldsymbol{\theta}$ is by maximizing the marginal log-likelihood $\log L$ of the multivariate Gaussian distribution over the training data set $(\boldsymbol{X}, \boldsymbol{y})$:

$$\begin{aligned} \log L(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\beta}) = & -\frac{1}{2} \boldsymbol{y}' K(\boldsymbol{X}, \boldsymbol{X})^{-1} \boldsymbol{y} \\ & - \frac{1}{2} \log\left[\det(K(\boldsymbol{X}, \boldsymbol{X}))\right] + \text{constant}. \end{aligned} \tag{19}$$

Adding a non-zero mean to the model is possible using explicit basis functions (De Spiegeleer *et al.* 2018). Therefore, replace the Gaussian process $f \sim GP(0, k)$ with zero mean by

$$g(\boldsymbol{x}) = f(\boldsymbol{x}) + h(\boldsymbol{x})' \boldsymbol{\beta}, \tag{20}$$

with $h(\boldsymbol{x})$ a set of fixed basis functions and corresponding coefficient set $\boldsymbol{\beta}$. Frequently implemented sets of basis functions are the linear basis $h(\boldsymbol{x}) = (1, \boldsymbol{x})$ and the quadratic basis $h(\boldsymbol{x}) = (1, \boldsymbol{x}, \boldsymbol{x}^2)$. If a non-zero mean is introduced using explicit basis functions, the parameter sets $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ are jointly estimated using the aforementioned maximum likelihood method.

3.2. Tree-based machine learning

The fundamental component of a tree-based machine learning method in regression is the regression tree (Breiman *et al.* 1984). Regression trees partition the predictor space in non-overlapping regions, predicting the same value $\hat{y}_{R_j}$ for each member of the constructed subregions $\{R_j \mid j = 1, \ldots, J\}$:

$$f_{\text{tree}}(\boldsymbol{x}) = \sum_{j=1}^{J} \hat{y}_{R_j} \cdot \mathbb{1}_{(\boldsymbol{x} \in R_j)}. \tag{21}$$

For a continuous response variable, $\hat{y}_{R_j}$ is often taken as the mean response value over all training observations in region R_j.

Regression trees are constructed using the CART algorithm (Breiman *et al.* 1984). Essentially, the algorithm selects, at each node, the feature and accompanying split point such that the updated predictions after the split result in the largest decrease of a tailored loss function $\mathcal{L}$. This process is iterated until a predetermined stopping criterion is met. As per the assumptions underlying a Gaussian process regression model, we define the loss function as the mean squared error of prediction:

$$\mathcal{L}(y, f(\boldsymbol{x})) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2. \tag{22}$$

A regression tree on its own is however not a powerful predictor, seeing that it is sensitive to changes in the data and prone

to overfitting. To improve the predictive accuracy, regression trees are therefore often aggregated in an ensemble of trees. We consider two ensemble methods: the random forest and gradient boosting machine.

3.2.1. Random forest. Random forest is an ensemble learning method initially introduced by Breiman (2001), which employs the bagging or bootstrap aggregating technique (Breiman 1996). Starting from the set of observations $D = (\boldsymbol{X}, y) = \{(\boldsymbol{x}_i, y_i) \mid i = 1, \ldots, n\}$, the bagging approach combines T regression trees, each of which is built on a different bootstrapped subset D_t of size $\delta \cdot n$, $0 < \delta < 1$. The final prediction from the bagging model is obtained by aggregating the predictions of the individual trees, taking the average value:

$$f_{\text{bagging}}(\boldsymbol{x}) = \frac{1}{T} \sum_{t=1}^{T} f_{\text{tree}}(\boldsymbol{x} \mid D_t). \tag{23}$$

To decorrelate the individual trees, random forest adds feature randomness to the bagging procedure. At each split, m candidate features for splitting are randomly selected out of the complete set of features, which is of size p. The random forest predictions $f_{\text{rf}}(\boldsymbol{x})$ are made based on Equation (23), while incorporating the aforementioned adaptation.

3.2.2. Gradient boosting machine. In contrast to the independent construction of trees in a random forest model, a gradient boosting machine employs an iterative procedure wherein T trees are grown sequentially during the training of the model (Friedman 2001). At each iteration t, a small tree of depth d is built, learning from predecessors, on a subsample of pseudo-residuals $\rho_{i,t}$, for observation i, of the loss function $\mathcal{L}$

$$\rho_{i,t} = -\left[\frac{\partial \mathcal{L}(y_i, f(\boldsymbol{x}_i))}{\partial f(\boldsymbol{x}_i)}\right]_{f=f_{t-1}}. \tag{24}$$

The use of pseudo-residuals, indicating the direction of steepest loss function reduction, enables the identification of areas in which the current fit of the model exhibits the greatest error. By selectively targeting these regions, the tree at iteration t is able to improve the model's overall performance. Subsampling a portion, with size $\delta \cdot n$ and $0 < \delta < 1$, of the pseudo-residuals has been demonstrated to enhance both the efficiency of the training process and the predictive accuracy of the model (Friedman 2002).

For each created region $\{R_{j,t} \mid j = 1, \ldots, J_t\}$, at iteration t, we calculate

$$\hat{b}_{j,t} = \operatorname{argmin}_b \sum_{i:\, \boldsymbol{x}_i \in R_{j,t}} \mathcal{L}\left(y_i, f_{t-1}(\boldsymbol{x}_i) + b\right), \tag{25}$$

to update the current fit to

$$f_t(\boldsymbol{x}_i) = f_{t-1}(\boldsymbol{x}_i) + \lambda \sum_{j=1}^{J_t} \hat{b}_{j,t} \cdot \mathbb{1}_{(\boldsymbol{x} \in R_{j,t})}. \tag{26}$$

The shrinkage parameter λ controls the learning speed of the algorithm. The process iterates until t reaches the predetermined number of trees T. The final prediction $f_{\text{gbm}}(\boldsymbol{x})$ corresponds to $f_T(\boldsymbol{x})$, which is obtained using Equation (26).

3.3. *Purged walk-forward validation*

Cross-validation is a widely used technique to tune the parameters of a machine learning algorithm, so as to optimize its performance and prevent it from overfitting. Moreover, this technique allows for the determination of the generalization error of the algorithm on multiple data sets. To this intent, the available data observations are split in subsets of 3 different kinds. A training set is used to build the model, a validation set to tune the parameters and a test set to evaluate the out-of-sample performance of the final model with optimal parameters. An overview of the parameters of the aforementioned models is provided in Table 6. The parameters for which a set of possible values is provided, are considered as tuning parameters. An extensive grid search is performed to find the optimal values for these tuning parameters, i.e. the values that minimize the out-of-sample prediction error on the validation set. The remaining model parameters are fixed at a sensible value. Note that for the Gaussian process regression model, a choice for the basis and kernel function is optimized using the validation set, not the parameter sets defining these functions.

The standard application of cross-validation is based on the assumption that all observations are drawn from an i.i.d. process. Each observation is then randomly (potentially using stratified random sampling) assigned to a unique subset, being it a training, validation or test set. However, the i.i.d. assumption is violated in many financial data sets. Specifically in this work, we observe prices of the same contract over time, which results in large serial correlation between different observations. Also, in both type of data sets, we observe the price of different contracts at the same date, meaning that the pricing of these contracts is subject to the same market conditions, creating again a dependence structure. In addition, random assignment of observations does not stroke with reality, since fair predictions can only be made without information on the future. Therefore, we examine purged, walk-forward validation schemes to account for the temporal dependence in

Table 6. Overview of the parameters of a Gaussian process regression model, random forest and gradient boosting machine.

GPR	Basis function	h	{Linear, Quadratic}
	Kernel function	k	{Squared Exponential, Matern32, Matern52[ab]}
RF	Trees	T	$\{100, \ldots, 3000\}$
	Split candidates	m	$\{\sqrt{p}, p-2\}$
	Sample size	δ	0.75
GBM	Trees	T	$\{100, \ldots, 3000\}$
	Learning speed	λ	0.01
	Tree depth	d	$\{1, 3, 5, 7\}$
	Sample size	δ	0.75

Note: If a set of possible values is given, the parameter is a tuning parameter, optimized in a validation setting.
[a]The kernel functions use a separate length-scale parameter for each feature, as for instance introduced in Equation (18).
[b]The Matern32 kernel fixes a parameter ν=1.5 for the modified Bessel function in Equation (17). Equivalently, Matern52 uses ν=2.5.

Figure 4. Graphical representation of a 3-fold purged, walk-forward validation scheme, with variable training length.

Table 7. Number of folds used in the validation framework.

	Type I			Type II	
	S&P 500	AAPL	JPM	Equity	Index
Folds	5	3	3	2	1

financial data (de Prado 2018). This scheme is applied to time-ordered data and relies on arranging observations in a way that preserves their temporal ordering.

Figure 4 provides a graphical representation of a 3-fold validation scheme, where the training set typically consists of a number of contiguous blocks and different folds are created by *walking forward* in time. The data is partitioned in blocks that cover a time period of approximately the same length. In order to prevent information leakage across the training, validation and test sets, *purging* is employed. Specifically, observations that occur at the same point in time are assigned to the same set, to ensure data is not shared across the partitions. To enhance the purged, walk-forward validation scheme, we introduce the ability to vary the length of the training window. More specifically, the starting point of the training window is optimized as an additional parameter using the validation set. We hereby eliminate the need for human intervention in training data selection. Moreover, the algorithm can exclude older observations that may not be as informative, thereby limiting their impact on model performance.

4. Evaluating the pricing performance

We evaluate the pricing performance of the different algorithms of Section 3, combined with the input settings discussed in Section 2.2.3, within a purged, walk-forward validation framework. The number of folds is tailored to the size of the data sets, with fewer folds implemented when the amount of available observations is relatively low. Table 7 outlines the technical details.

The mean absolute error (MAE) of prediction is used as a performance measure:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - f(\boldsymbol{x}_i)|\,, \tag{27}$$

with y_i the observed value for `IVOL` (type I) or `Strike` (type II) and $f(\boldsymbol{x}_i)$ the predicted value. This way, by the nature of the response variables, the average distance between the true value and the estimation can be quantified in volatility points. To demonstrate the predictive power of the machine learning techniques, we also include the pricing error for a simple linear regression (LR) model, for each data set and input setting, of the form

$$f_{LR}(\boldsymbol{x}_i) = \beta_0 + \boldsymbol{\beta}\boldsymbol{x}_i, \tag{28}$$

with coefficient set $\{\beta_0, \boldsymbol{\beta}\}$, and $\boldsymbol{\beta}$ of size p, the number of input features.

4.1. Data type I: running volatility swaps

Figures 5–7 compare the out-of-sample pricing performance of the different algorithms, for each of six input settings, across the different data sets of type I. We evaluate the mean absolute error of prediction on the test set of each data fold and report on the average value (left) and fold-specific value (right). The fold-specific values are limited to those obtained using the optimal algorithm for prediction. The models are trained using a flexible training window, as introduced in Section 3.3. In Section A, we elaborate on the improvements of using a flexible training window, compared to a fixed window, based on the S&P 500 data set.

A number of conclusions can be drawn from the presented results. First, the optimal algorithm for prediction varies depending on the underlying asset, though it is clear that the machine learning techniques outperform the linear regression model. In particular, on the S&P 500 data set, the Gaussian process regression model outperforms the tree-based machine learning methods. The opposite is true for the data sets on AAPL and JPM, where the random forest algorithm is emerging as the best performer. However, it is worth noting that the computation time of the gradient boosting machine and random forest model are, respectively, about 10 times and about 7 times higher compared to that of a Gaussian process regression.

Second, the inclusion of a skewness-based feature leads to a notable improvement in the predictions, for market-implied moments (S2) in general and implied volatilities (S5), when using GPR. Consistently across the data sets, the optimal prediction algorithm achieves the highest predictive accuracy by combining information on volatility, skewness and kurtosis in the input features of setting S6. The improvement in performance over settings S4 and S5 is however minimal for the

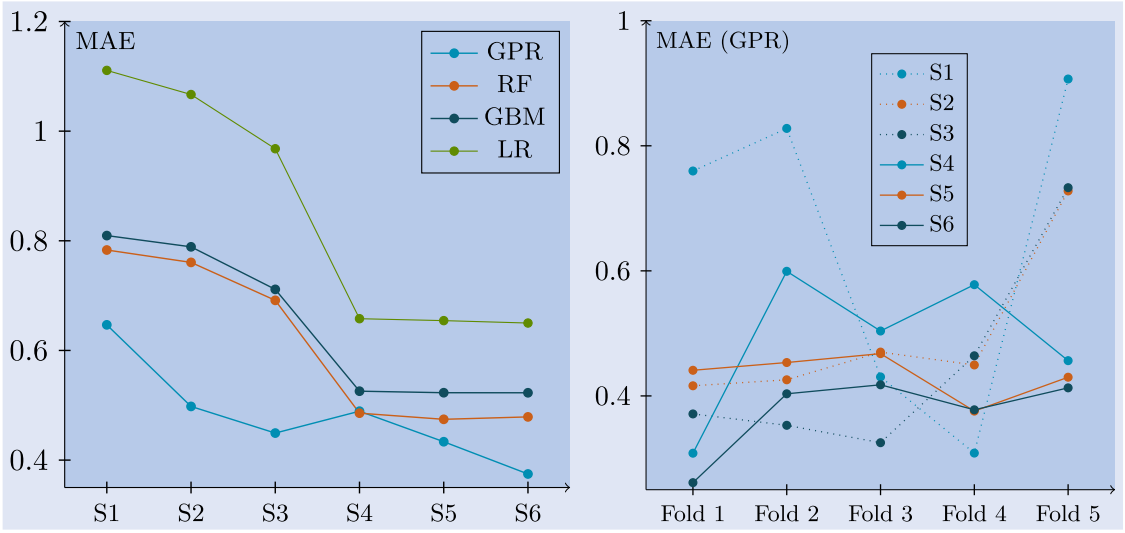

Figure 5. Average (left) and fold-specific (right) out-of-sample performance on the S&P 500 data set. The optimal GPR algorithm is used on the right.

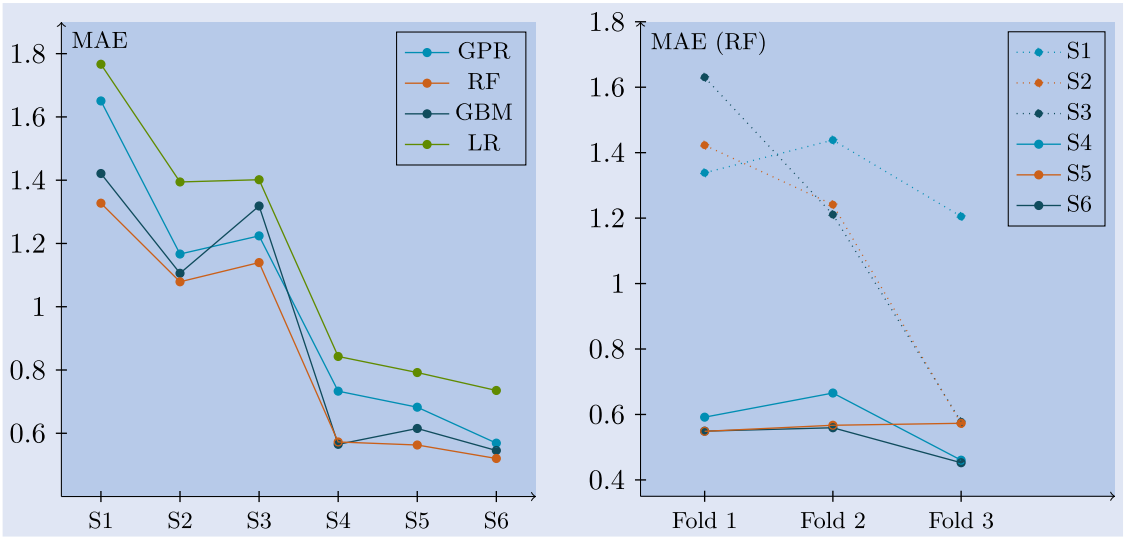

Figure 6. Average (left) and fold-specific (right) out-of-sample performance on the AAPL data set. The optimal RF algorithm is used on the right.

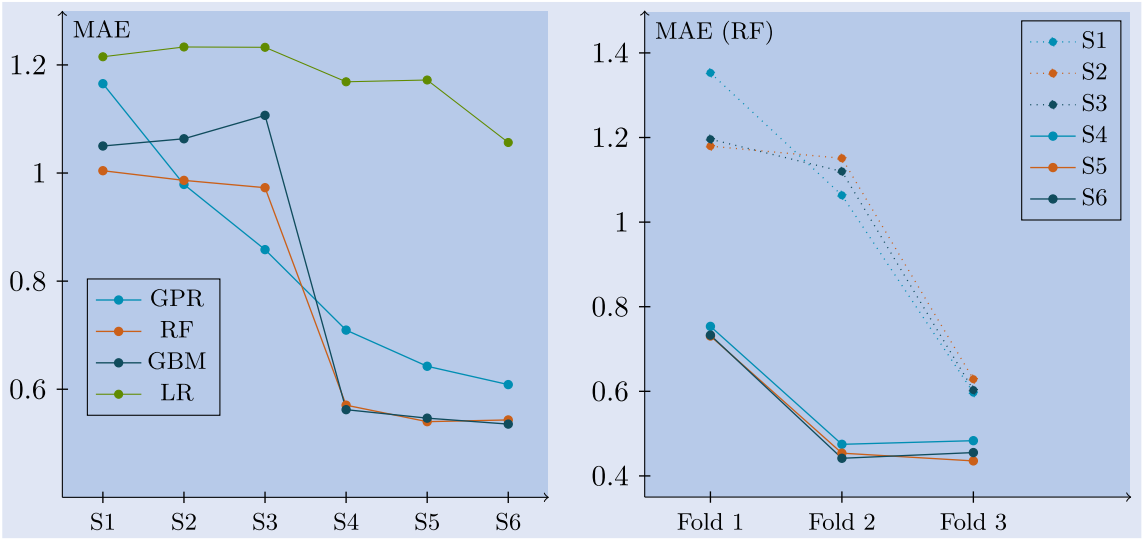

Figure 7. Average (left) and fold-specific (right) out-of-sample performance on the JPM data set. The optimal RF algorithm is used on the right.

AAPL and JPM data set. In general, the input settings using implied volatility features demonstrate clear superior performance in comparison to the settings using market-implied features.

Last, we note significant variations (jumps) in the performance of the market-implied settings S1 to S3 when evaluating the out-of-sample error on each test fold, using the optimal algorithm for prediction. In contrast, the implied volatility-based input settings show more stable outcomes. Incorporating features based on skewness and kurtosis in setting S6 consistently improves the predictive accuracy compared to S3, whereas the inclusion of only the kurtosis feature in setting S5 does not have a similar impact.

Figure 8 is used to visually assess the out-of-sample performance of a GPR model with input features of setting S4 to S6, on the S&P 500 data set. Figure 8(a), which displays the predicted values for `IVOL` against the observed values, shows a distinct cluster of points stretched along the unit line, with the most outlying observations highlighted in orange. Further examination reveals that these points are observed very close to the maturity date of the contract, with an average value of the variable `Weight` equal to 0.96.

Next, we compare the pricing performance of implied volatility-based input settings with respect to specific ranges of the variable `Weight`. For this purpose, we partition the out-of-sample data based on the time remaining until maturity and calculate the mean absolute error of prediction for each partition. Our findings, presented in Figure 8(b), first reveal a notable increase in prediction error for observations approaching the maturity of the contract, which aligns with the identified outliers in Figure 8(a). Second, we observe that, while the performance of the different settings is comparable for observations with longer remaining time till maturity, higher-order features significantly enhance predictive accuracy within the last category.

Figure 8(c) presents the out-of-sample mean absolute error, using implied volatility-based input features, for different ranges of `IVOL`, hereby isolating extreme observations. While the predictive performance of the three settings is comparable for mid-values of `IVOL`, it is observed that employing the input configuration specified in setting S6 significantly outperforms settings S4 and S5 in the outer ranges. This finding indicates that the inclusion of higher-order information regarding the underlying asset is valuable in achieving more accurate predictions for non-standard values of `IVOL`.

We conclude this part by leveraging the estimated values for `IVOL` to predict the contract's price `PX`, using Equation (5). We present the average out-of-sample results on the data sets of type I in Figure 9(a). For the estimation of `IVOL`, we use the optimal algorithm combined with the input features of setting S6. Additionally, we show the mean absolute error of prediction, when directly estimating `PX` using the aforementioned modeling settings. The results are

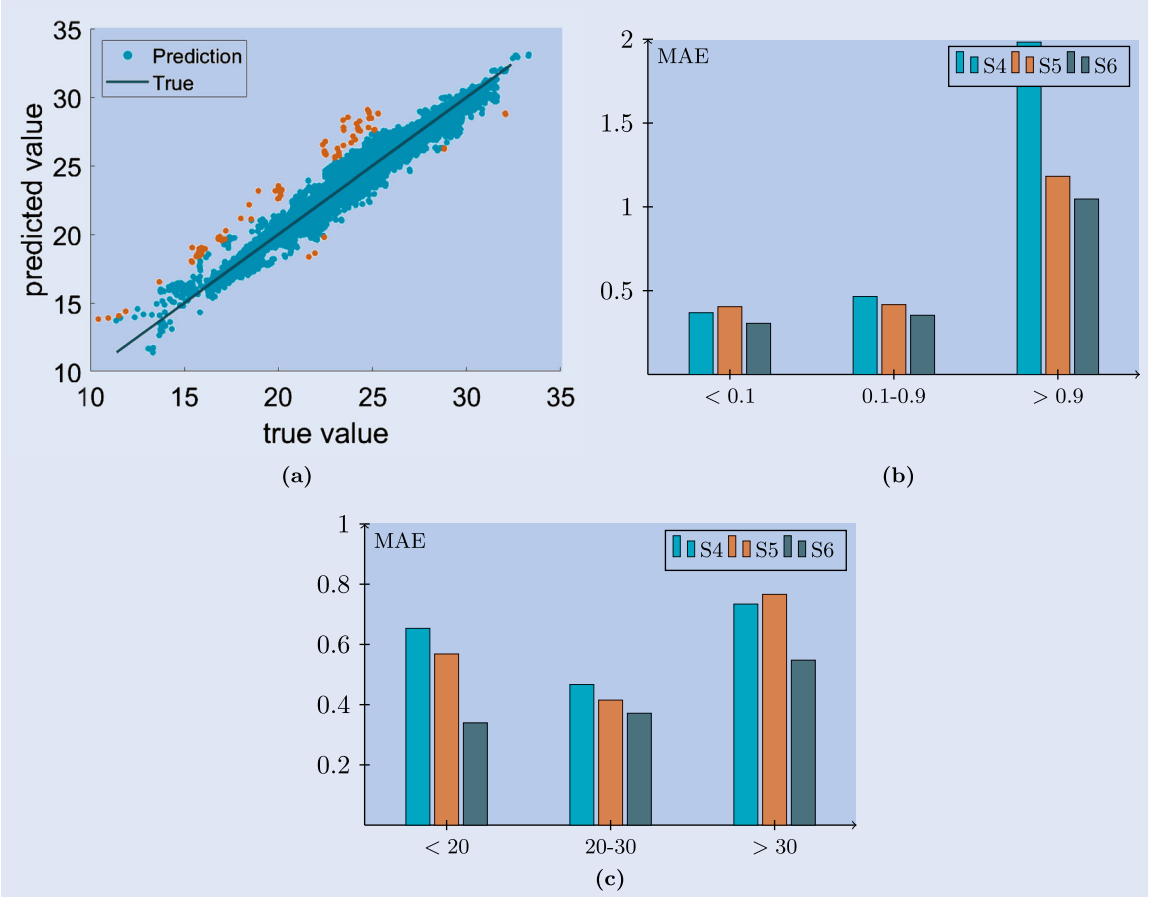

Figure 8. (a) Scatter plot of the true value of `IVOL` against the predicted value under S6. The out-of-sample MAE of prediction within different ranges of (b) `Weight` and (c) `IVOL`. All calculations are based on the S&P 500 data set.

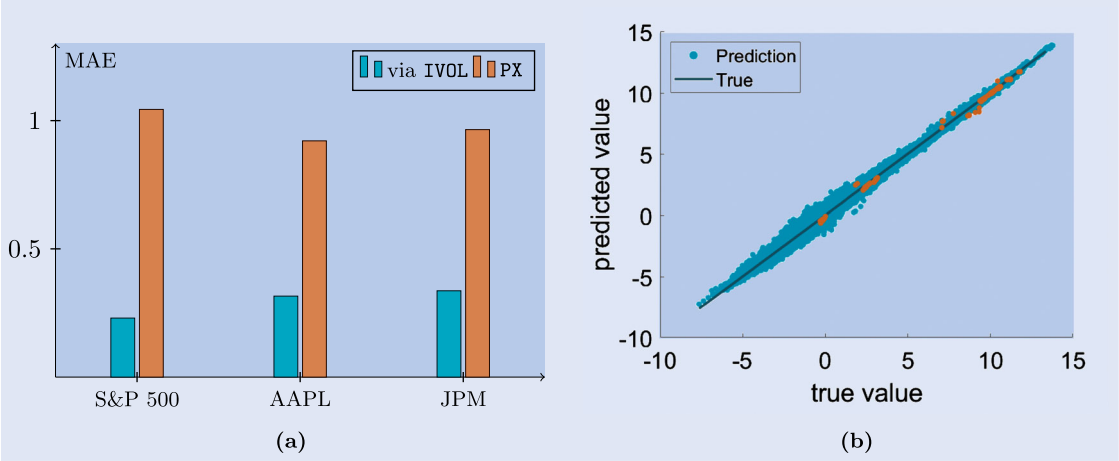

Figure 9. (a) Comparison of the out-of-sample MAE for PX, with and without using the introduction of IVOL. (b) Scatter plot of the true value of PX against the predicted value under S6, using a GPR model on the S&P 500 data set.

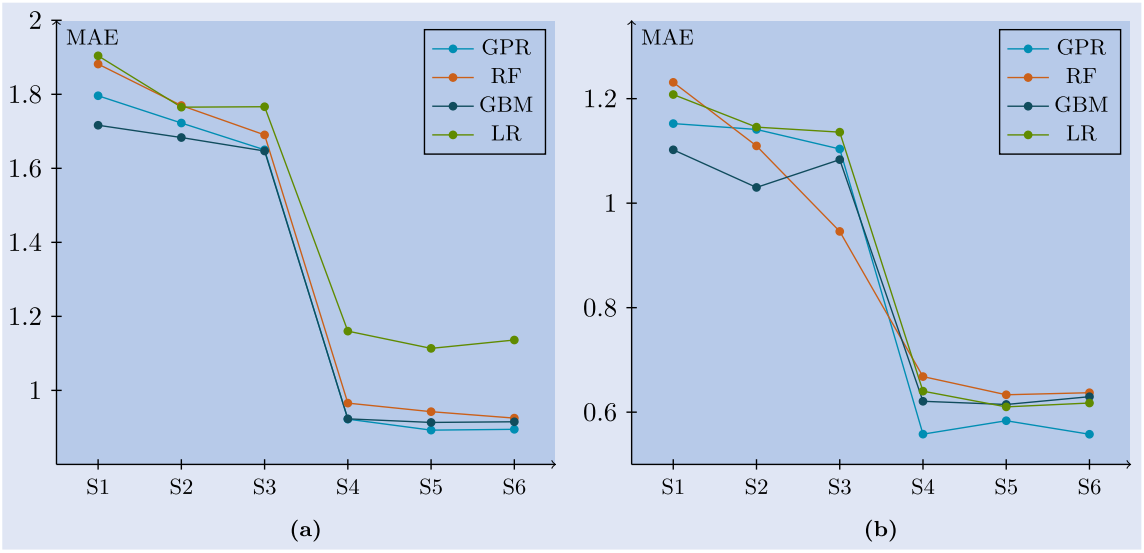

Figure 10. Out-of-sample performance on the (a) Equity and (b) Index data sets of type II of a GPR, RF and GBM model, for different input settings.

respectively shown in blue and orange and confirm the importance of introducing the IVOL parameter.

To accompany the results in Figure 9(a) on the S&P 500 index, we present the optimally predicted values for PX in comparison to the true prices in Figure 9(b). It is observed that, as the observation date approaches the contract maturity, the variable Weight approximates unity, and the predicted value of IVOL has minimal impact on the contract price. Consequently, the previously identified outlying observations in Figure 8(a) no longer appear as such in the scatter plot of Figure 9(b), and PX is estimated with high precision.

4.2. *Data type II: volatility swaps at initiation*

Figure 10(a,b) compare the average out-of-sample pricing performance of the different algorithms, for each of six input settings, across the data sets of type II. Consistent with the results discussed in Section 4.1, we observe that a decision on the optimal input setting is independent of the selected algorithm. Although the differences in performance are minor, the Gaussian process regression model outperforms the tree-based machine learning methods in both data sets. The input settings using implied volatility-based features again demonstrate clear superior performance. In line with the results presented in Figure 8(b), the inclusion of higher-order features does not lead to a notable improvement in the predictions, for prices observed at maximal time distance from the contract's maturity date.

In general, when comparing the pricing performance across the two types of data sets, a slightly higher mean absolute error appears for the data sets of type II. This result may be attributed to a few factors. First, the observations in the data sets of type II are associated with contracts written on various underlying assets, whereas type I data sets focus solely on a single asset. It is reasonable to assume that an asset's own pricing history is the most informative. Second, the size of the data sets of type II is significantly smaller. Increasing the number of training observations could enhance the pricing quality of our models. For the smallest data set, the data set

on indices in particular, additional data is especially needed for the machine learning methods to be able to distinguish in performance from a linear model.

5. Conclusion

This paper demonstrates the use of data-driven, machine learning techniques in the pricing of capped volatility swaps. Given the significant dependence of the price of the instrument on fluctuations in the underlying asset, we explore distributional information on the asset as a potential source for predictive features. To that intend, the forward-looking, market-implied moments of an asset's pricing-return density are estimated from quoted European vanilla options. However, the calculations of the presented input variables, especially market-implied skewness and kurtosis, show substantial day-to-day instabilities. Subsequently, forward implied volatilities at various moneyness levels are proposed to construct alternative measures of volatility, skewness and kurtosis. In general, with the inclusion of higher-order features, we aim to capture the effect of the cap level that is inherent in the price of a volatility swap and only reached when extreme movements in the underlying asset are observed.

Tree-based machine learning models and a Gaussian process regression model are considered as possible candidate algorithms for predictions. A tailored purged walk-forward validation setting for financial data is used to tune the parameters of the algorithms and to determine their out-of-sample pricing ability. A training data selection is automatically performed by allowing the training window to have a variable length.

From the presented analysis, we conclude that the incorporation of higher-order features improves the models' predictive accuracy, regardless of the utilized algorithm and input source, but especially for outer ranges of the response variables, and observations close to the maturity date of the contract. Consistently across the different data sets, we observe that using the implied volatility-based input settings is a secure strategy. The performance on a specific data set is however dependent on the chosen algorithm. While the Gaussian process regression is preferred on both the S&P 500 data set and the data sets of type II, the random forest model outperforms the Gaussian process regression model on the type I data sets of individual stocks.

Through this research, we establish the foundations of a functional pricing tool for capped volatility swaps that utilizes the product's market prices as its primary data source and does not require a workable expression for the highly non-linear pricing structure of the instrument. Moreover, an estimation of the price can potentially be delivered at any point during the life time of the product.

Disclosure statement

No potential conflict of interest was reported by the author(s).

References

Acworth, P.A., Broadie, M. and Glasserman, P., A comparison of some monte carlo and quasi monte carlo techniques for option pricing. In *Monte Carlo and Quasi-Monte Carlo Methods 1996*, edited by H. Niederreiter, P. Hellekalek, G. Larcher, and P. Zinterhof, pp. 1–18, 1998 (Springer New York: New York, NY).

Bakshi, G., Kapadia, N. and Madan, D., Stock return characteristics, skew laws, and the differential pricing of individual equity options. *Rev. Financ. Stud.*, 2003, **16**(1), 101–143.

Bloomberg, Introduction into the new bloomberg implied volatility calculations. Technical document, 2008. Available online at: https://pdfslide.net/documents/bloomberg-implied-volatility-method-2008-new.html?page=1,

Breiman, L., Bagging predictors. *Mach. Learn.*, 1996, **24**(2), 123–140.

Breiman, L., Random forests. *Mach. Learn.*, 2001, **45**(1), 5–32.

Breiman, L., Friedman, J., Stone, C. and Olshen, R., *Classification and Regression Trees*, 1984 (Taylor & Francis Group: New York).

Brockhaus, O. and Long, D., Volatility swaps made simple. *Risk*, 2000, **1**, 92–95.

Carr, P. and Lee, R., Volatility derivatives. *Annu. Rev. Financial Econ.*, 2009, **1**(1), 319–339.

Carr, P. and Madan, D., Towards a theory of volatility trading. In *Option Pricing, Interest Rates, and Risk Management*, edited by E. Jouini, J. Cvitanic, and M. Musiela, pp. 458–476, 2001 (Cambridge University Press: Cambridge, UK).

Davis, J., Devos, L., Reyners, S. and Schoutens, W., Gradient boosting for quantitative finance. *J. Comput. Finance*, 2020, **24**(4), 1–40.

de Prado, M.L., *Advances in Financial Machine Learning*, 2018 (John Wiley & Sons, Inc.: Hoboken, New Jersey).

De Spiegeleer, J., Madan, D., Reyners, S. and Schoutens, W., Machine learning for quantitative finance: Fast derivative pricing, hedging and fitting. *Quant. Finance*, 2018, **18**(10), 1635–1643.

Demeterfi, K., Derman, E., Kamal, M. and Zou, J., More than you ever wanted to know about volatility swaps. In *Quantitative Strategies Research Notes*, 1999 (Goldman Sachs: New York).

Flavell, R.R., *Swaps and Other Derivatives*, 2010 (John Wiley & Sons, Inc.: Hoboken, New Jersey).

Friedman, J., Greedy function approximation: A gradient boosting machine. *Ann. Stat.*, 2001, **29**(5), 1189–1232.

Friedman, J., Stochastic gradient boosting. *Comput. Stat. Data. Anal.*, 2002, **38**(4), 367–378.

Gan, L., Wang, H. and Yang, Z., Machine learning solutions to challenges in finance: An application to the pricing of financial products. *Technol. Forecast. Soc. Change*, 2020, **153**, 119928.

Guillaume, F. and Schoutens, W., Calibration risk: Illustrating the impact of calibration risk under the heston model. *Rev. Deriv. Res.*, 2012, **15**(1), 57–79.

Harrison, M. and Pliska, S., Martingales and stochastic integrals in the theory of continuous trading. *Stoch. Process. Their Appl.*, 1981, **11**(3), 215–260.

He, X.-J. and Zhu, S.-P., Analytical pricing formulae for variance and volatility swaps with a new stochastic volatility and interest rate model. *Expert Syst. Appl.*, 2022, **206**, 117880.

Issaka, A., Variance swaps, volatility swaps, hedging and bounds under multi-factor heston stochastic volatility model. *Stoch. Anal. Appl.*, 2020, **38**(5), 856–874.

Kim, S.-W. and Kim, J.-H., Volatility and variance swaps and options in the fractional sabr model. *Eur. J. Finance*, 2020, **26**(17), 1725–1745.

Lin, S. and He, X.-J., Pricing variance and volatility swaps with stochastic volatility, stochastic interest rate and regime switching. *Phys. A*, 2020, **537**, 122714.

Madan, D. and Schoutens, W., *Applied Conic Finance*, 2016 (Cambridge University Press: Cambridge, UK).

Mixon, S., What does implied volatility skew measure? *J. Deriv.*, 2011, **18**(4), 9–25.

Rasmussen, C. and Williams, C., *Gaussian Processes for Machine Learning*, 2006 (MIT Press: Cambridge, MA).
Ross, S.A., The arbitrage theory of capital asset pricing. *J. Econ. Theory*, 1976, **13**(3), 341–360.
Rujivan, S. and Rakwongwan, U., Analytically pricing volatility swaps and volatility options with discrete sampling: Nonlinear payoff volatility derivatives. *Commun. Nonlinear Sci. Numer. Simul.*, 2021, **100**, 105849.
Salmon, N. and SenGupta, I., Fractional barndorff-nielsen and shephard model: Applications in variance and volatility swaps, and hedging. *Ann. Financ.*, 2021, **17**(4), 529–558.

Appendix. Flexible training window

Table A1 provides an overview of the mean absolute error of prediction when fitting the Gaussian process regression model to the S&P 500 data set. We report on the average error over the five data folds, for each of six different input settings. If the model is trained using a fixed training window, the start of this window is fixed at the first observation in the data set, for each data fold. We observe that using a flexible training window consistently improves the model's fit, with on average, a decrease in MAE equal to 0.11.

Table A1. MAE of prediction for a Gaussian process regression model on the S&P 500 data set of type I.

	S1	S2	S3	S4	S5	S6
Fixed	0.7574	0.5901	0.5844	0.5662	0.5855	0.4913
Flexible	0.6467	0.4979	0.4493	0.4892	0.4335	0.3747

Note: The model is trained with both a fixed and flexible training window.