Data Science in Finance and Economics

https://www.aimspress.com/journal/dsfe

DSFE, 6(1): 85–120.
DOI:10.3934/DSFE.2026004
Received: 19 March 2025
Revised: 18 December 2025
Accepted: 20 January 2026
Published: 27 February 2026

Research article

Identifying market dynamics through the Hurst exponent

Fabrizio Di Sciorio*, Laura Molero González and Juan E. Trinidad-Segovia

Department of Economics and Business, University of Almería, 04120, Almería, Spain

* **Correspondence:** Email: fd940@inlumine.ual.es.

Abstract: In this study, we analyze a historical time series of crude oil prices to estimate the realized volatility and the Hurst exponent, with the objective of characterizing the temporal dynamics and persistence of the market. A clustering framework is employed to categorize historical realized volatility into three distinct regimes: low, moderate, and high volatility, thereby labeling observations accordingly. Subsequently, multiple machine learning models are implemented to predict these volatility regimes using the Hurst exponent as a key feature. The predictive performance of different algorithms is systematically evaluated to assess the effectiveness of the Hurst exponent as a volatility indicator. The findings suggest that the Hurst exponent provides a robust relative measure of market turbulence, demonstrating its potential as a predictive metric for financial market dynamics.

Keywords: market efficiency; clustering; turbulent market period; classification model; market dynamics; Hurst exponent

JEL Codes: C22, C45, C53, G17

1. Introduction

Financial markets are complex systems characterized by volatile price movements driven by economic, political, and psychological factors. Understanding and forecasting market behavior is crucial for risk management, portfolio optimization, and trading strategies. Volatility, a key indicator of market stability, plays a central role in asset pricing and financial decision-making (Bašta and Molnár, 2018).

A crucial theoretical foundation for this work lies in the Efficient Market Hypothesis (EMH), which posits that asset prices fully reflect available information (Fama, 1970). Within this framework, the Hurst exponent provides a direct proxy for market efficiency: values close to 0.5 indicate a random walk behavior consistent with market efficiency, whereas deviations from $\frac{1}{2}$ ($H > \frac{1}{2}$ or $H < \frac{1}{2}$) imply, respectively, persistence (momentum) or anti-persistence (mean-reverting tendencies), both of which are symptomatic of inefficiencies. This theoretical linkage justifies our focus on the Hurst exponent as more than a statistical artifact — it becomes an interpretive measure of how oil markets transition

between efficient and inefficient phases over time.

Traditional models such as Autoregressive Conditional Heteroskedasticity (ARCH) and Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) analyze volatility dynamics, while advances in computational techniques have enabled machine learning methods for improved predictions, as explored by Cho et al. (Cho and Lee, 2022). The Hurst exponent is a measure of long-range dependence that provides insights into market behavior by distinguishing persistent trends, random walks, and mean-reverting patterns. Studies by Kristoufek (Kristoufek and Vosvrda, 2014a, 2016) highlight the effectiveness of the Hurst exponent in identifying volatile periods.

Alvarez-Ramirez (2002) analyzed daily crude oil prices using multi-fractal methods, showing that the market exhibits persistence and long-term memory effects. Their height-height correlation analysis reveals multi-fractal structures with mixed Hurst exponents, identifying two characteristic time scales—weeks and quarters—where price dynamics were extracted using moving-average filtering. These findings suggest that the crude oil market follows a random-walk assumption only at short time scales.

Mensi and Managi (2014) conducted an analysis of weak-form efficiency and structural breaks in crude oil benchmarks (WTI and Brent) from 1990 to 2012, highlighting the effectiveness of the Hurst exponent. Their study found that compared to Shannon entropy, the Hurst exponent better detects financial crises and extreme events, such as wars and terrorist attacks, offering valuable insights for risk managers and commodity portfolio hedgers.

Yao and Ju (2020) applied multi-fractal methods to analyze the nonlinear correlations among economic policy uncertainty (EPU), the crude oil market, and the stock market. Using multi-fractal detrended fluctuation analysis (MF-DFA), they found that each of the three series exhibits multifractality due to long-range correlations and fat-tailed distributions. Additionally, multi-fractal detrended cross-correlation analysis (MF-DCCA) revealed significant cross-correlation multi-fractal features among the three series, with the stock and oil price series showing the strongest correlation values. Their coupling detrended fluctuation analysis (CDFA) indicated that stock market fluctuations contribute more to the coupling correlation than crude oil or EPU. Finally, their multiscale multi-fractal analysis (MMA) visualized the dynamic behaviors of these correlations, demonstrating that the stock-oil price relationship maintains symmetry across different scales.

Fernández-Martínez (2017) compared three different approaches to calculating the Hurst exponent to detect early transitions from efficient market behavior to market bubbles. Their study demonstrated that self-similarity exponents serve as indicators of herding behavior in financial markets. By analyzing S&P500 stocks, they suggested that a higher self-similarity exponent correlates with increased price persistence, helping to anticipate market bubbles before they fully develop. Their research evaluated the performance of the Generalized Hurst Exponent, Detrended Fluctuation Analysis, and GM2 algorithms in identifying nascent bubbles, contributing to the broader understanding of financial stability and market efficiency.

This study, moving in a theoretical framework where prices follow a multi-fractional motion, estimates the realized volatility and Hurst exponent (time varying) in the oil price time series to capture the market persistence. This paper contributes to the existing literature by investigating the explanatory power of the time-varying Hurst exponent for volatility regimes in oil markets, following the approach of Bianchi and Mattera (2022). While prior research has extensively examined volatility modeling using traditional econometric and machine learning techniques, few studies have explored the role of long-memory and persistence—captured through a time-varying Hurst exponent—in explaining shifts

between volatility regimes. This work aims to fill that gap by adopting a multi-fractional Brownian motion (mBm) framework, which allows for dynamic local roughness in the price process.

Specifically, the study estimates both realized volatility and the Hurst exponent on oil price time series to capture changes in market persistence.

The integration of statistical and machine learning methods in this study is motivated by the dual need to measure underlying market structures and to predict future states. While statistical tools provide interpretable metrics (such as Hurst values) that are theoretically grounded, machine learning models leverage these features for regime classification, thereby bridging descriptive and predictive approaches within a coherent analytical framework.

Using the K-means algorithm, we identify distinct volatility regimes—low, moderate, and high—and employ machine learning models to assess whether the Hurst exponent can anticipate or explain these phases. The best-performing models are validated on an independent test set to ensure robustness.

The paper is structured as follows: Section 2 presents the theoretical background on multi-fractional Brownian motion and its relevance for financial modeling. Sections 3 and 4 outline the methods used for estimating the Hurst exponent and realized volatility. Section 5 provides the empirical results, and section 6 concludes with key findings and implications.

2. Multi-fractional Brownian motion

A significant advancement in the study of fractional Brownian motion is the introduction of multi-fractional Brownian motion (mBm). The traditional fractional Brownian motion (fBm) assumes a constant Hurst exponent H, which can be unrealistic as it does not account for the time-varying behavior of investors. This constant H implies a uniform investment behavior throughout different phases of the economic cycle, see Mattera et al. (2022).

Initially introduced by Peltier and Levy-Vehel (1995) and later expanded upon by researchers such as Bianchi (2005), multifractional Brownian motion (mBm) extends the traditional fractional Brownian motion framework by allowing the Hurst exponent to vary over time. This time-varying Hurst exponent provides a dynamic lens through which changes in market efficiency can be interpreted. Periods in which $H(t)$ deviates from $\frac{1}{2}$ signal structural inefficiencies in the market, whereas intervals where $H(t) \approx \frac{1}{2}$ suggest a tendency toward informational efficiency.

The concept of multi-fractional Brownian motion was developed to address the limitations of fBm. The core idea is to replace the fixed Hurst exponent H with a function $H(t)$ that varies over time, where $H(t)$ is constrained to the interval $[0, 1]$ and serves as the regularity function of the mBm. Began with the definition of a Fractional Brownian Field. Consider a probability space (Ω, F, P). A fractional Brownian field on $\mathbb{R} \times [0, 1]$ is defined as a Gaussian field $(W(t, H))_{(t,H)\in\mathbb{R}\times[0,1]}$, where, for each H in $[0, 1]$, the process behaves as a fractional Brownian motion characterized by the Hurst parameter H. A multi-fractional Brownian motion can then be understood as a "path" traced out within this fractional Brownian field. Specifically, if $h : \mathbb{R} \to [0, 1]$ is a continuous deterministic function and W represents a fractional Brownian field, then the mBm corresponding to W with functional parameter H is defined by the Gaussian process $W_t^H := W(t, H(t))$ for all $t \in \mathbb{R}$. The multi-fractional Brownian motion can be expressed by the following equation (Peltier and Levy-Vehel, 1995):

$$W_{H(t)}(t) = \int_{\mathbb{R}} (t-s)_+^{H(t)-\frac{1}{2}} - (-s)_+^{H(t)-\frac{1}{2}} dW_s, \tag{1}$$

where $(s)_+$ denotes the positive part of the function. Additionally, the covariance of the multi-fractional Brownian motion is given by (Bianchi and Pianese, 2014):

$$\mathbb{E}[W_{H(t)}(t)W_{H(s)}(s)] = K^2 D(H(t), H(s))(t^{H(t)} + s^{H(s)} - (t-s)^{H(t)+H(s)}), \tag{2}$$

where:

$$D(H(t), H(s)) = \frac{\sqrt{\Gamma(2H(t)+1)\Gamma(2H(s)+1)\sin(\pi H(t))\sin(\pi H(s))}}{2\Gamma(2H(t)+H(s)+1)\sin\left[\pi\left(\frac{H(t)+H(s)}{2}\right)\right]}. \tag{3}$$

Building on the representation by Carmona et al. (2003), developed a stochastic integral, leading to the expression $W_H = \int_0^t K_H(t,s)dW_s$. To approximate W_H as a semi-martingale, they applied a kernel smoothing technique.

Overall, the Hurst exponent is intrinsically linked to market efficiency. Several researchers, including Bianchi (2005) and Granero and et al. (2008), have demonstrated that a Hurst exponent value of $\frac{1}{2}$ indicates an efficient market.

Consequently, when developing a measure of inefficiency, it is essential to evaluate the deviations of the Hurst exponent H from the value $\frac{1}{2}$ (Kristoufek and Vosvrda, 2014b).

The aim of this paper is to utilize the Hurst exponent as a key driver in identifying crises, with an application to the oil market. Specifically, starting from volatility clusters identified through a clustering model the three market phases are: moderate volatility, high volatility, and low volatility. On this, a supervised classification model is employed.

3. Estimation of time-varying Hurst exponent

The Hurst exponent has become an excellent measure for long memory of financial time series. It has been later proposed for use in fractal analysis (Mandelbrot and Wallis, 1969; Mandelbrot, 1982). Lately, fractal analysis has become popular in financial research, which extended the fractal analysis to economic financial dynamics (Mattera et al., 2022).

Numerous methods exist in the literature to estimate the Hurst exponent see (Granero and et al., 2008). One classical approach is R/S analysis, which involves calculating the rescaled range at various time scales and examining the scaling behavior. By plotting the rescaled range against the time scale, one can derive an estimate of the Hurst exponent.

Another method, the variance method, takes advantage of the self-similarity properties of the fractional Brownian motion (fBm) to estimate the Hurst exponent based on the variance equation (Di Sciorio and Segovia, 2023). Wavelet transform-based techniques, such as Continuous Wavelet Transform (CWT) and Wavelet Leader-Based Scaling (WLBS), assess the scaling properties of wavelet coefficients to estimate the Hurst exponent. In our work, we employ the AMBE method introduced by Bianchi and Pianese.

The AMBE method, first defined as an extension of Pèltier and Lévy Véhel's work on pointwise regularity, has been developed specifically to estimate the local Hölder regularity $\alpha_X(t)$ of a time series. Its mathematical formulation leverages the absolute moments of increments over moving windows of size δ, yielding a locally adaptive measure of regularity.

The estimator has several properties that make it particularly suitable for our study:

- handles time series with regime shifts and structural breaks—features typical of crude oil prices—better than classical variance-based or wavelet methods.
- is asymptotically normally distributed, and its variance can be explicitly derived, enabling formal statistical inference. Optimal parameters ($q = 1, k = 2$) minimize this variance, ensuring estimation stability even for relatively short time windows.
- unlike more computationally intensive wavelet-based estimators, allows efficient rolling-window computation, essential for a multi-decade dataset (2000–2024) without compromising accuracy.

Given these advantages, it provides an ideal balance of theoretical rigor, computational feasibility, and suitability for financial time series analysis, justifying its selection for this study over alternative Hurst exponent estimators like the variance method or wavelet-based approaches.

Given the path $X_{h_t}(t)$ produced by multi-fractional Brownian motion (mBm), we define a rolling window of size δ and a lag q. The method utilizes discrete sampling $\{X_{i,n}\}_{i=1,\dots,n-1}$, where, for each i, j ranges from $i - \delta$ to $i - q$. The parameters are set with $i = \delta + 1, \dots, n$ and $q = 1, \dots, \delta$. At time t_0, the mBm behaves like a fractional Brownian motion (fBm) with an exponent $h(t_0)$, allowing us to define the AMBE estimator as follows:

$$\hat{h}_t = -\frac{\log\left(\sqrt{\pi}\mathcal{S}^k / \left(2^{k/2}\Gamma\left(\frac{k+1}{2}\right)K^k\right)\right)}{k\log\left(\frac{n+1}{q}\right)} \tag{4}$$

In this equation, $\mathcal{S}^k$ is defined as:

$$\mathcal{S}^k = \frac{1}{\delta - q + 1}\sum_{j=i-\delta}^{i-q}\left|X_{j+q,n} - X_{j,n}\right|^k, \quad i = \delta + 1, \dots, n$$

The optimal parameter selections that minimize the variance of the estimator are $q = 1$ and $k = 2$. The window size δ should be chosen to balance the variance of the estimator and the promptness of the signal response. For financial applications, Bianchi and Pianese recommend setting $\delta = 30$.

To ensure full reproducibility, detailed implementation guidelines, pseudo-code, and executable examples (Python) are provided in the A.

To evaluate and compare the performance of different Hurst exponent estimators, we conducted a series of Monte Carlo simulations using synthetic time series generated as fractional Brownian motions (fBm) with prescribed Hurst exponents $H \in \{0.2, 0.5, 0.7, 0.9\}$. For each true Hurst exponent, $n_{\text{sim}} = 10000$ independent realizations of length $N = 5000$ were generated. Each realization serves as a testbed for three estimation methods: AMBE, R/S analysis, variance method.

For the AMBE method, we followed the formulation outlined in Equation (4), using a rolling window of size $\delta = 30$ and lag $q = 1$. For the R/S analysis, the rescaled range was computed across multiple

window sizes to obtain a scaling relation whose slope provides an estimate of H. The variance method leveraged the self-similarity property of fBm by computing the standard deviation of increments at varying lags and regressing their logarithm against the logarithm of the lag.

The simulation procedure allows us to systematically compare the statistical properties of the estimators, including their mean, median, and standard deviation. Boxplots (Figure 1) of the estimated Hurst exponents versus the true values were produced to visually assess bias and dispersion. This setup provides a controlled framework to validate the accuracy and robustness of the AMBE estimator relative to classical methods, highlighting its potential advantages in capturing local regularity in financial time series.

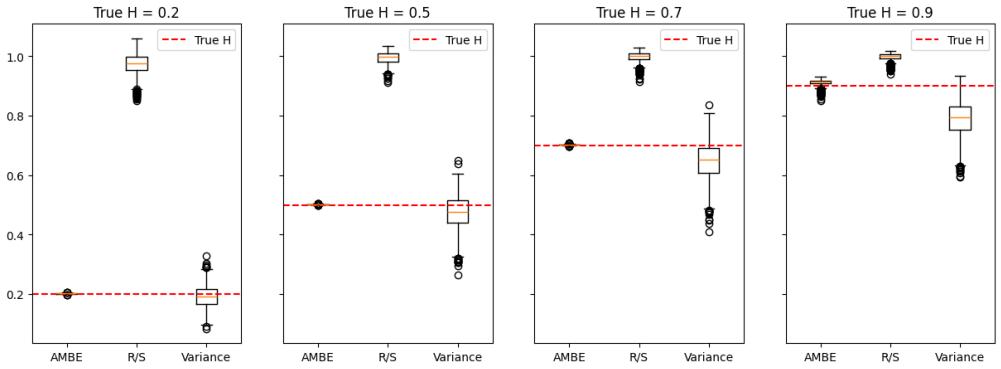
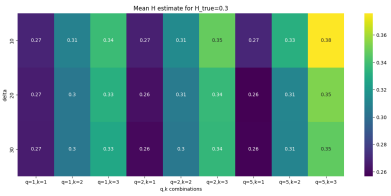
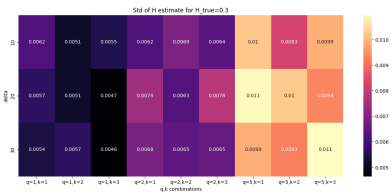


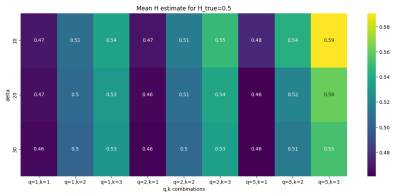
Figure 1. Comparison methods for H estimation.



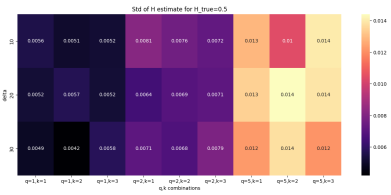
(a) Mean estimated $\hat{H}$ when the theoretical $H = 0.3$



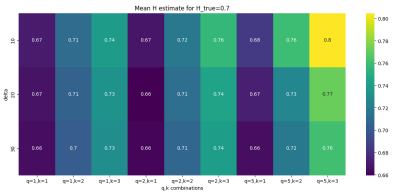
(b) Variance estimated $\hat{H}$ when the theoretical $H = 0.3$



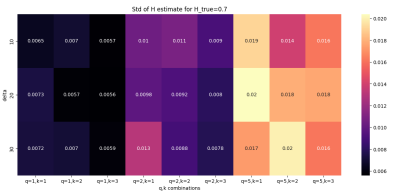
(c) Mean estimated $\hat{H}$ using AMBE when the theoretical $H = 0.5$



(d) Variance estimated $\hat{H}$ when the theoretical $H = 0.5$



(e) Mean estimated $\hat{H}$ using AMBE when the theoretical $H = 0.7$



(f) Variance estimated $\hat{H}$ when the theoretical $H = 0.7$

Figure 2. Comparison of different Hurst estimation methods across multiple q, k, δ, scenarios.

A sensitivity analysis (Figure 2) was conducted to evaluate the impact of varying k, q, and δ on the estimated Hurst exponent under different market conditions: $H = 0.3$, indicative of a mean-reverting market; $H = 0.5$, representing an efficient market; and $H = 0.7$, suggesting a trending market. The

analysis involved computing the mean and variance of the estimated Hurst exponents across multiple simulations for each parameter combination. The results indicate that the AMBE method's performance is influenced by the choice of parameters, with the optimal combination of $\delta = 30$, $k = 2$, and $q = 1$ providing stable estimates across all market conditions, in line with the recommendations of the original authors.

4. Estimation of realized volatility

Realized volatility is a fundamental concept in financial econometrics, measuring the extent of asset return fluctuations over a given period. Unlike implied volatility, which is derived from option prices, realized volatility is calculated directly from historical return data. As an empirical measure, it plays a crucial role in risk management, derivative pricing, and portfolio optimization by capturing actual market dynamics.

In the fractional framework, a first adjustment to the classical realized volatility was made by Chen and Pigorsch (2010), and consists in modifying how squared returns are aggregated. The expected realized volatility is follows:

$$\mathbb{E}[RV_T] \propto T^{2H-1}. \tag{5}$$

Thus, a corrected estimator for fractional volatility can be defined as:

$$RV_T(H) = T^{1-2H} \sum_{t=1}^{T} r_t^2. \tag{6}$$

This adjustment compensates for the distortion introduced by the factor T^{2H-1}, thereby normalizing the realized volatility relative to time. In our implementation, returns are defined as daily log-returns:

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right)$$

based on adjusted closing prices. Realized volatility is computed over a rolling window of 30 trading days by summing squared returns within the window. From a theoretical perspective, market efficiency can be viewed as a particular case within the broader framework of fractional markets. Specifically, when the Hurst exponent H equals $\frac{1}{2}$, the auto-covariance function of fractional Brownian motion coincides with that of standard Brownian motion. In this sense, the Hurst exponent provides a quantitative measure to assess how closely a financial market behaves according to the Efficient Market Hypothesis (EMH). By analyzing the time-varying Hurst exponent $H(t)$, we can study the day-to-day deviations of a market from efficiency. Following Mattera et al. (2022), one can define a simple inefficiency measure as

$$\frac{1}{2} - H(t),$$

which captures the degree and direction of deviation from the efficient benchmark (Figure 3).

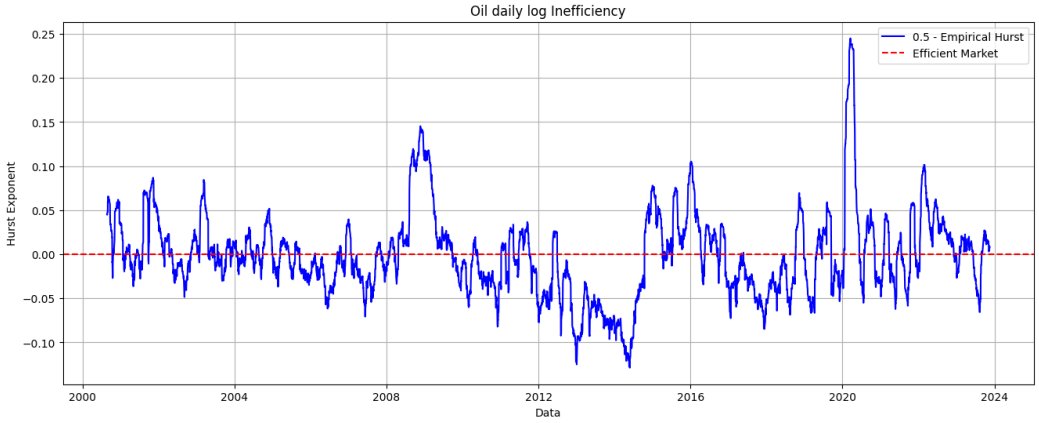


Figure 3. Inefficiency vs volatility.

Empirical evidence (Figure 4) shows that during certain crisis periods, there is an inverse relationship between $H(t)$ and volatility, consistent with findings in the literature. For instance, periods where the market exhibits H significantly different from $\frac{1}{2}$ indicate persistent trends ($H > \frac{1}{2}$) or mean-reverting behavior ($H < \frac{1}{2}$), signaling inefficiency.

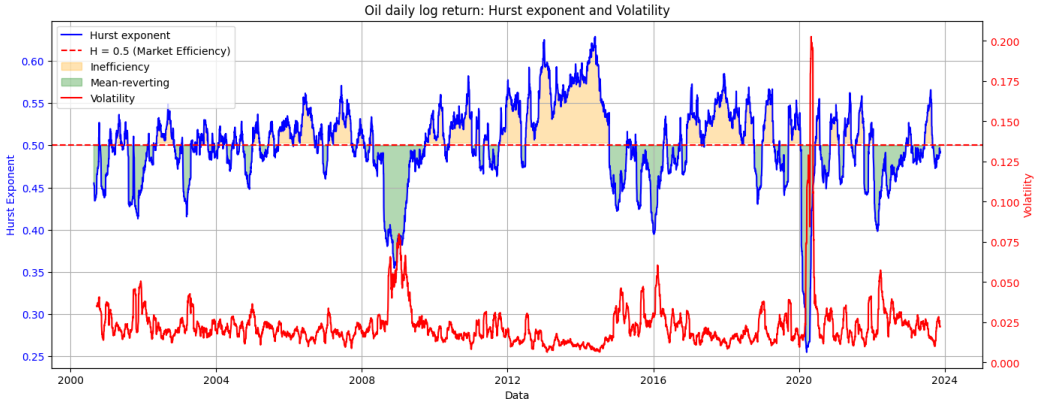


Figure 4. Inefficiency estimation.

In practice, these deviations can have direct implications for trading strategies and risk management. Persistent trends ($H > \frac{1}{2}$) may allow for momentum-based strategies, whereas mean-reverting periods ($H < \frac{1}{2}$) suggest potential for contrarian strategies. Moreover, the degree of inefficiency quantified by $\frac{1}{2} - H(t)$ can be used to adjust risk exposure, informing portfolio allocation, hedging decisions, and volatility targeting.

In this way, the Hurst exponent not only provides a theoretical bridge between fractional market models and classical EMH but also offers actionable insights into market dynamics, inefficiencies, and practical risk management over time.

5. Empirical analysis

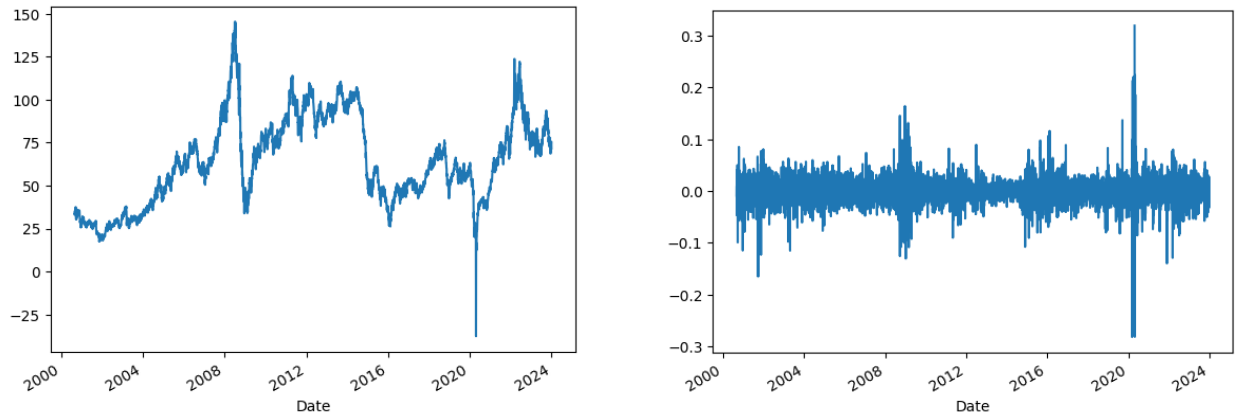
5.1. Data

To achieve the goals of this study, the adjusted daily closing prices of crude oil have been considered. The study period covers from January 1st, 2000 to January 1st, 2024, totaling 5863 observations. Table 1 presents the descriptive statistics of the oil adjusted price, the realized volatility, the Hurst exponent, and the logarithmic returns. It summarizes the key characteristics of the data.

Table 1. Descriptive statistics.

	Oil Adjusted Price	Realized Volatility	Hurst	Log. Return
Mean	64.210988	0.022337	0.522391	0.000233
St. Deviation	25.460900	0.011683	0.065821	0.026310
Min.	-37.630001	0.006392	0.000000	-0.282206
25%	44.852500	0.015661	0.502352	-0.012846
50%	62.154999	0.019765	0.529674	0.001096
75%	83.757502	0.025383	0.554206	0.013556
Max.	145.289993	0.128964	0.653665	0.319634

The average oil-adjusted price is approximately 64.21, indicating relative stability, with a standard deviation of 25.46 reflecting significant price fluctuations. Prices range from -37.63 to 145.29, showing extreme lows and highs due to external factors. The average volatility is 0.0223, with a standard deviation of 0.0117, indicating moderate variations. Volatility ranges from 0.0064 to 0.1290, suggesting periods of stability followed by sharp increases. The Hurst exponent averages 0.5224, close to 0.5, implying near-random price behavior. With a deviation of 0.0658, values show limited variation, ranging from 0 to 0.6537, indicating occasional persistence. The average logarithmic return is 0.000233, suggesting minor price changes. A standard deviation of 0.0263 shows some variability, with returns spanning from -0.2822 to 0.3196, reflecting substantial daily swings. Figures 5a and 5b shows the evolution of the crude oil price and its logarithmic return for the study period considered in the study.



(a) **(b)**

Figure 5. Evolution of (a) the Adjusted Price and (b) the Logarithmic Return of Crude Oil for the period from 2000 to 2024.

To check the stationarity of the time series considered in this study, the Dickey-Fuller test is employed. Table 2 shows the results. As can be seen, time series are stationary for both the estimation of volatility and the estimation of Hurst.

Table 2. Results from the Dickey-Fuller test.

	Volatility	Hurst
ADF Statistic	-7.257848	-6.982981
p-value	0.000000	0.000000
Critical Value 1%	-3.431	-3.431
Critical Value 5%	-2.862	-2.862
Critical Value 10%	-2.567	-2.567
Stationarity	Yes	Yes

5.2. Clustering analysis

The K-means and the Gaussian mixture model (GMM) algorithms (Di Sciorio, 2023) are used to determine the clusters by analyzing the patterns of realized volatility. As is known, K-means is an unsupervised clustering algorithm that is used to group data into a predefined number of groups, called clusters. The goal of K-means is to minimize intra-cluster variance, making the clusters as homogeneous as possible. In this study, we selected three clusters to identify distinct volatility regimes. The choice of three clusters is motivated by both theoretical and empirical considerations: financial markets typically exhibit phases of low, moderate, and high volatility, each associated with different market dynamics and investor behaviors. This tripartite classification aligns with previous literature on volatility regime detection and allows for a meaningful segmentation of market conditions. Three clusters where identified:

- **Moderate-Volatility Phase:** this cluster represents a transition phase where volatility is moderate. During these periods, markets may be influenced by external events, changes in economic policies, or significant variations in fundamental data. Volatility in this state is higher than in the stable-trend phase but does not reach the peaks observed during crises. Traders and investors can use this phase to adjust their strategies, anticipating potential future movements.
- **High-Volatility Phase:** this cluster is characterized by pronounced price fluctuations, indicating a phase of heightened market instability. Such conditions often arise due to macroeconomic shocks, geopolitical tensions, supply-demand imbalances, or speculative activity. While these periods may coincide with financial crises, they primarily reflect volatility spikes rather than systemic economic breakdowns. Traders exercise increased caution, and predictive models must adapt to rapid and unpredictable price swings.
- **Low-Volatility:** this cluster corresponds to a period of market stability, where price movements are relatively subdued. Prices tend to follow a more predictable trajectory with minimal fluctuations, creating a favorable environment for long-term investment strategies. In this phase, predictive models often perform more effectively, as market relationships remain more consistent and less disrupted by external shocks.

Table 3 shows the results on test set of the cluster analysis performed. The results show a satisfactory fit with a cluster goodness index (silhouette score) of 0.60. In addition, the low values of the Davies-Bouldin index are obtained, meaning that clusters are compact and well-separated.

Although the silhouette score of 0.60 indicates a reasonable cluster structure, it also suggests that the separation between clusters is moderate rather than sharp. This implies that the identified groupings primarily capture volatility states, while transitions in market efficiency may occur more gradually and overlap between regimes, rather than forming perfectly distinct phases. Table 4 presents the centroids of the different realized volatility clusters. In algorithms such as K-means, centroids are used to represent a cluster and serve as a reference point during the clustering process.

Table 3. Clustering analysis results.

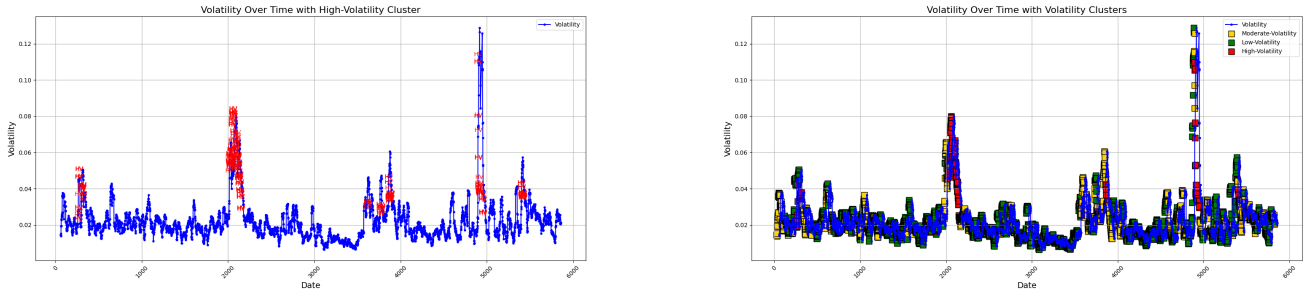
Method	Silhouette Score	Davies-Bouldin Index
K-Means	0.60	0.58
GMM	0.56	0.58

Table 4. Cluster centroids for realized volatility.

Market Phases	Centroid
Moderate volatility	0.028383
High volatility	0.059981
Low volatility	0.016381

Figures 6a and 6b shows the clusters of volatility in relation to time and Hurst. There exists an inverse relationship, noted by the literature, between Hurst and volatility, see Angelini and Bianchi (2023).

Our results confirm that during high-volatility phases, the Hurst exponent deviates further from $\frac{1}{2}$, suggesting that market turbulence is often accompanied by lower informational efficiency. This reinforces the interpretation of volatility regimes not merely as changes in price fluctuation intensity, but as reflections of dynamic shifts in market efficiency. As shown in Figure 2, as volatility increases, $H(t)$ deviates from $\frac{1}{2}$, reaching lower values. This behavior is consistent with findings in the financial literature Angelini and Bianchi (2023), supporting the idea that high-volatility periods are associated with decreased market efficiency.



(a) Volatility time series with HV cluster.

(b) Volatility time series with volatility clusters.

Figure 6. Volatility with clustering over time.

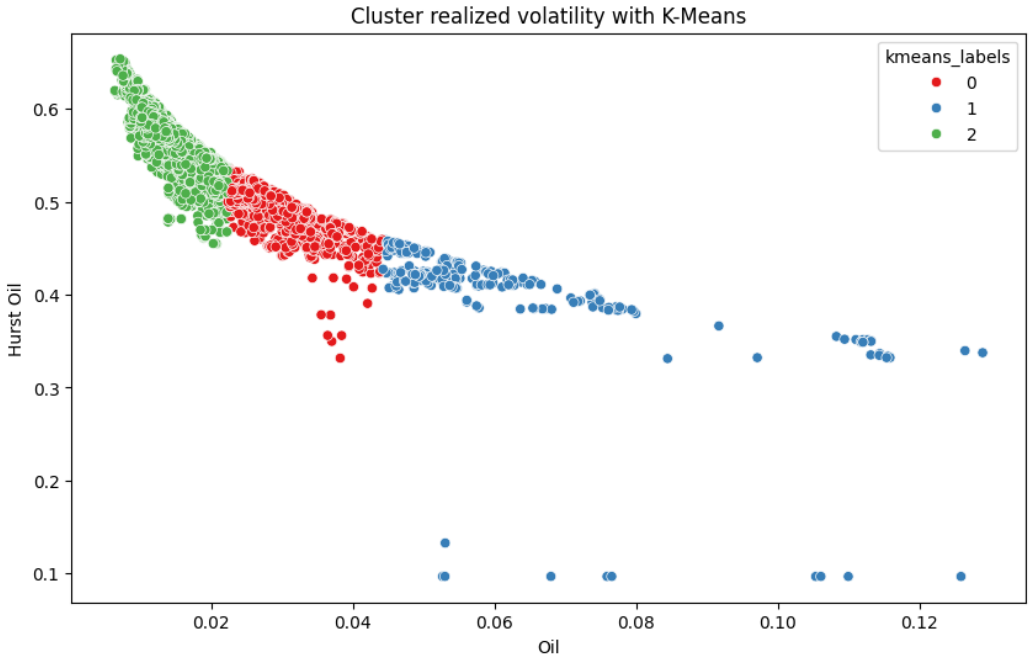


Figure 7. Cluster volatility by k-means.

Once the observations have been clustered (Figure 7) based on their realized volatility, the dataset is ready for use in a supervised learning problem, specifically a classification task. The primary objective is to employ the Hurst exponent as a predictor for the identified volatility clusters. To achieve this, we proceed with the analysis of both traditional statistical models and machine learning algorithms, aiming to evaluate their capacity to model the relationship between the Hurst exponent and the realized volatility regimes. The clustering process provides distinct volatility regimes that serve as class labels for the classification problem.

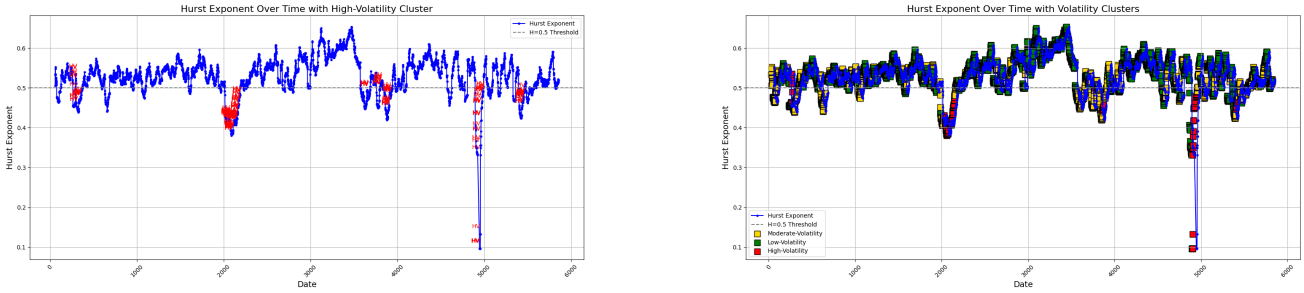
To avoid look-ahead bias, clustering algorithms (K-means/GMM) were always fit on the training set only, or within each rolling training window, and subsequently applied to validation and test sets using the fixed centroids derived from the training data. As a robustness check, we repeated the analysis by restricting clustering to training data only and confirmed that the resulting volatility regime labels and predictive performance remained virtually unchanged.

Furthermore, clustering validation metrics, such as the silhouette score and Davies-Bouldin index (reported in Table 3), confirm that three clusters provide a good balance between cluster compactness and separation.

By integrating the Hurst exponent as the key feature, we investigated its potential as a discriminative variable between the different volatility clusters (Figure 8b). Various models, including logistic regression, decision trees, and ensemble methods, as well as more advanced machine learning approaches like support vector machines and neural networks, are implemented and compared to assess their predictive performance.

This analysis aims to identify the most effective approach for leveraging the Hurst exponent to predict volatility states, providing insight into its utility in financial market forecasting and its integration within volatility models. In the literature, tree-based models, such as decision trees and random forests, face challenges in handling time series data due to their inability to capture the temporal dimension

of information. These models cannot directly utilize the temporal sequence of events, which can compromise their ability to make accurate predictions in contexts where the dependent variable is influenced by past values.



(a) Hurst exponent time series with HV cluster.

(b) Hurst exponent time series with volatility clusters.

Figure 8. Hurst exponent with clustering over time.

To overcome this limitation, it is possible to introduce lag features, which represent the previous values of the variable of interest. This strategy allows tree-based models to incorporate additional information that reflects the temporal dynamics of the time series.

Moreover, boosting models also encounter difficulties in managing time series data. Although these models are more advanced and flexible than simple decision trees, their ability to handle temporal aspects is limited for similar reasons. They also do not automatically consider the order of events, which means that, without proper data preparation, they may overlook significant temporal relationships.

To improve the performance of boosting models on temporal data, similar strategies can be adopted:

- Integration of lag features. Adding lag variables can help capture temporal dynamics. Selecting significant lags (for instance, through ACF or PACF analysis) is crucial.
- Feature Engineering. In addition to lags, including other temporal characteristics such as trends, seasonality, and holidays can provide a richer context for the model.
- Temporal Cross-validation. Utilizing a temporal cross-validation strategy helps ensure that the model is not trained on future data, providing a more realistic estimate of its performance.

The number of lag features to include in the model was determined through an analysis of the autocorrelation function (ACF) of the Hurst time series. Specifically, the first 30 significant lags were selected to ensure that the models could adequately capture the relationships between past and present data, thereby enhancing their predictive capability. This approach enabled a structured integration of temporal variables, increasing the robustness of the model and facilitating more accurate predictions. Figure 9 shows the autocorrelation function and the partial autocorrelation function used to define significant time lags of Hurst in the classification model. The time lags of Hurst were used as features in the tree models.

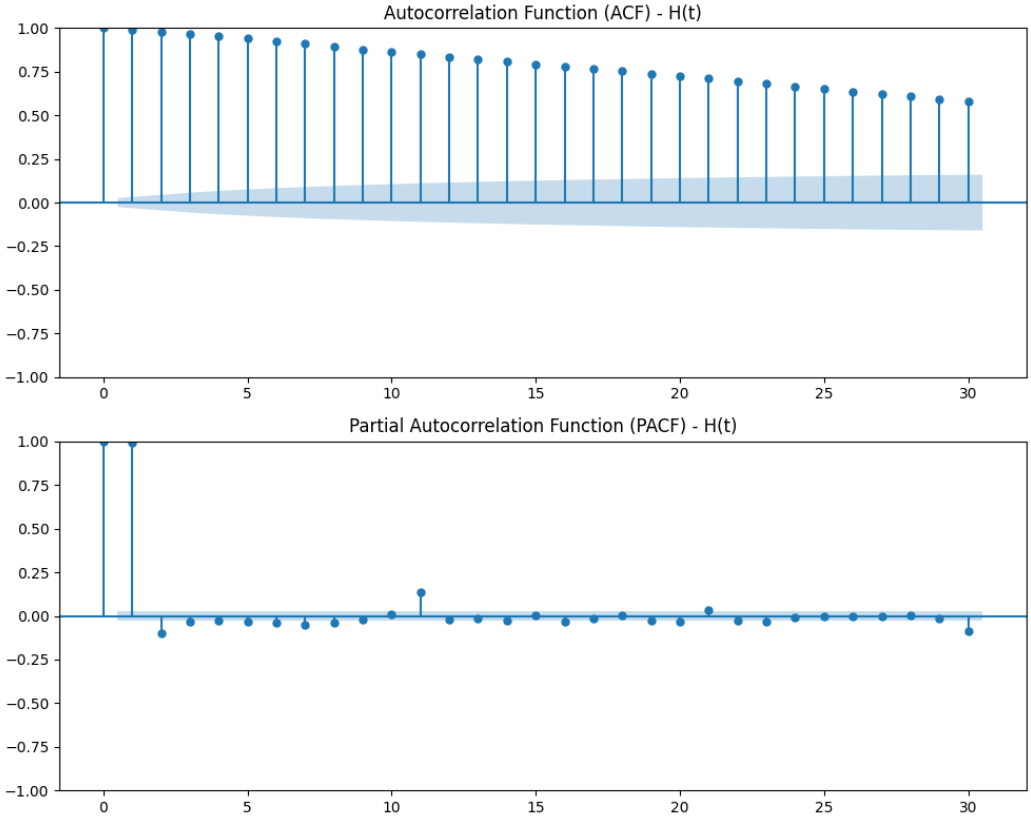


Figure 9. Autocorrelation function (ACF) and partial autocorrelation function (PACF) of $H(t)$.

To achieve the objective of this study, the database was divided into a train (80%), from 11/2000 to 04/2019, and a test set (20%), from 05/2019 to 12/2023; with shuffle parameters set *False*, in order to preserve the original order of the dataset. Following recent literature Chatzis (2018),Lin and Tsai (2011), Samitas and Kenourgios (2020) and Ahn (2011), various models have been tested to select the best classification model using a basic configuration of hyperparameters. Below is a summary of the models tested:

- **Logistic Regression**: a linear model based on the sigmoid function to map predictions to probabilities.
- **Random Forest**: a bagging method that constructs multiple decision trees and aggregates their predictions to improve accuracy and control overfitting.
- **XGBoost**: an advanced gradient boosting algorithm known for its high performance.
- **LightGBM**: a gradient boosting framework that uses tree-based learning algorithms, optimized for speed and memory efficiency, often outperforming other boosting methods.
- **SVM (Support Vector Machine)**: a classification algorithm that finds the optimal hyperplane to separate data into different classes, effective for high-dimensional data.
- **KNN (K-Nearest Neighbors)**: a learning algorithm that classifies new data points based on the majority class among its nearest neighbors.

- **Naive Bayes**: a probabilistic classifier based on Bayes' theorem, assuming independence between features. Works well with small datasets, but may struggle with feature correlation.
- **CatBoost**: a gradient boosting algorithm specifically designed to handle categorical features efficiently, reducing overfitting and improving prediction accuracy.

Below (table 5), we report the main characteristics of the machine learning models tested.

Table 5. Machine learning models tested: main hyper-parameters and key characteristics.

Model	Main hyper-parameters	Key Characteristics
Logistic Regression	`C`, `penalty`, `solver`	Linear model based on the sigmoid function; interpretable; suitable for linear classification tasks.
Random Forest	`n_estimators`, `max_depth`, `min_samples_split`	Bagging method: builds multiple decision trees; handles overfitting well.
XGBoost	`learning_rate`, `n_estimators`, `max_depth`, `subsample`	Gradient boosting: strong predictive performance; built-in regularization; highly configurable.
LightGBM	`num_leaves`, `learning_rate`, `max_depth`, `boosting_type`	Gradient boosting optimized for speed and memory efficiency; performs well on large, sparse datasets.
SVM	`C`, `kernel`, `gamma`	Robust classifier; effective in high-dimensional spaces; supports non-linear kernels.
KNN	`n_neighbors`, `metric`, `weights`	Instance-based learner; simple and intuitive; computationally intensive on large datasets.
Naive Bayes	`var_smoothing` (for GaussianNB)	Probabilistic classifier; fast and suitable for small datasets; assumes feature independence.
CatBoost	`iterations`, `learning_rate`, `depth`	Gradient boosting is designed to handle categorical features efficiently, reduces overfitting; high accuracy.

6. Results

To select the best model, a temporal split of the dataset was implemented with an 80–20 ratio. Additionally, a Bayesian optimization procedure using Optuna was applied, as described in detail in the current section. To further assess model robustness and avoid overfitting, a lag selection (feature selection) was performed. Moreover, for some models, such as XGBoost, CatBoost, and LightGBM, regularization parameters were also included in the hyperparameter search to further improve generalization.

Table 6 shows the results of the models on the test set.

Table 6. Classification Models Results.

Model	Moderate volatility	High volatility	Low volatility	Accuracy
	Precision / Recall / F1	Precision / Recall / F1	Precision / Recall / F1	
Logistic Regression	0.85 / 0.83 / 0.84	0.79 / 0.95 / 0.86	0.86 / 0.87 / 0.87	0.85
Random Forest	0.86 / 0.87 / 0.86	0.76 / 0.87 / 0.81	0.90 / 0.87 / 0.89	0.87
XGBoost	0.86 / 0.86 / 0.86	0.84 / 0.97 / 0.90	0.89 / 0.87 / 0.88	0.87
LightGBM	0.86 / 0.88 / 0.87	0.77 / 0.90 / 0.83	0.91 / 0.87 / 0.89	**0.88**
SVM	0.84 / 0.88 / 0.86	0.86 / 0.78 / 0.82	0.90 / 0.86 / 0.88	0.87
KNN	0.82 / 0.71 / 0.76	0.57 / 0.76 / 0.65	0.80 / 0.87 / 0.83	0.79
Naive Bayes	0.62 / 0.68 / 0.65	0.41 / 0.73 / 0.53	0.74 / 0.60 / 0.67	0.65
Catboost	0.86 / 0.89 / 0.88	0.79 / 0.71 / 0.75	0.91 / 0.89 / 0.90	**0.88**

The performance of various machine learning models in classifying market phases reveals interesting insights, particularly highlighting the strengths of lightGBM. This model achieved an impressive accuracy of 0.88, outperforming others in the evaluation. In the Transition phase, lightGBM displayed a precision of 0.86, recall of 0.88, and an F1-score of 0.87, indicating its effectiveness in identifying transitional market conditions. Notably, its performance in the Stable-trend phase was remarkable, with a precision of 0.91 and an F1-score of 0.89, showcasing its ability to accurately classify stable market periods. One of the critical advantages of lightGBM is its high recall of 0.90 in detecting crises, which is vital in financial markets for timely risk mitigation. However, it should be noted that its precision in the 'High-Volatility' phase was 0.77, suggesting a slight tendency for false positives. Compared to other models, such as logistic regression and random forest, lightGBM demonstrated a significant advantage in accuracy and robustness under different market conditions. It is evident that nonlinear models perform better because the relationship between Hurst and market volatility is not strictly linear; for further information see Angelini and Bianchi (2023).

The consistently high accuracy observed suggests the need to verify, in this section, the potential presence of overfitting, particularly in the presence of heterogeneous market regimes. To address this concern, we incorporate learning curve diagnostics and sensitivity analyses (e.g., varying lag structures and subsample testing) to ensure that the models capture genuine market dynamics rather than artifacts of the training data.

Once the best classification model was selected, Optuna (Akiba et al., 2019) was used for hyperparameter optimization. Optuna is an open-source hyperparameter optimization framework that leverages advanced search techniques such as Bayesian optimization to efficiently explore the hyper-parameter space. It is particularly suited for machine learning models, where selecting the appropriate hyperparameters has a significant impact on model performance. Optuna utilizes a *tree-structured parzen estimator* (TPE) approach for *Sequential Model-Based Optimization* (SMBO), which allows it to make informed decisions about the next set of hyperparameters to evaluate based on the results of previous trials.

The goal of hyper-parameter tuning in Optuna is to maximize (or minimize) a given objective function $f(\theta)$, where θ represents the set of hyper-parameters, and f is the evaluation metric of interest (in this case, accuracy). Mathematically, this is expressed as:

$$\theta^* = \arg\max_{\theta \in \Theta} f(\theta) \tag{7}$$

where Θ denotes the hyperparameter search space.

In the case of lightGBM, the objective function returns the accuracy of the model in the test set, which is defined as:

$$f(\theta) = \text{accuracy}(y_{\text{test}}, \hat{y}(\theta)) \tag{8}$$

where:

· $\hat{y}(\theta)$ are the predicted labels of the model using the hyper-parameters θ,
· y_{test} are the true labels from the test set,
· accuracy$(y_{\text{test}}, \hat{y})$ is the accuracy score of the model.

Optuna optimizes several key hyperparameters of the LightGBM classifier. Specifically:

- Number of estimators $\theta_{\text{n\_estimators}}$: the number of boosting iterations, which is suggested as an integer within the range $[10, 100]$,

$$\theta_{\text{n\_estimators}} \sim \mathcal{U}(10, 100) \tag{9}$$

- Learning rate $\theta_{\text{learning\_rate}}$: controls the step size in each iteration and is sampled from a log-uniform distribution within $[0.01, 0.1]$,

$$\theta_{\text{learning\_rate}} \sim \log \mathcal{U}(0.01, 0.1) \tag{10}$$

- Maximum tree depth $\theta_{\text{max\_depth}}$: this controls the maximum depth of the individual trees and is suggested as an integer within the range $[2, 7]$,

$$\theta_{\text{max\_depth}} \sim \mathcal{U}(2, 7) \tag{11}$$

These hyperparameters are sampled iteratively by Optuna, and the model's performance on the test set is evaluated after each trial.

Optuna employs Bayesian optimization with TPE to guide the search process. The TPE method constructs two probability density functions (PDFs) for the values of the objective function:

· $l(x)$, representing hyper-parameter values that lead to better model performance (higher accuracy),
· $g(x)$, representing the remaining hyper-parameter values.

At each iteration, the next set of hyper-parameters θ is chosen by maximizing the ratio:

$$\frac{l(x)}{g(x)} \tag{12}$$

This effectively directs the search towards the most promising regions of the hyper-parameter space, improving the efficiency of the optimization process. To ensure a rigorous evaluation of the model's

predictive capability, a validation strategy based on the rolling-window approach has been adopted. This method involves partitioning the time series into fixed-size windows, each consisting of a five-year training period ($T_{\text{train}} = 5$ years), followed by a one-year test period ($T_{\text{test}} = 1$ year). After each iteration, the window is shifted forward by one year while maintaining the same training set duration and updating the test set with the most recent data. The process is repeated until the entire available time horizon is covered.

Mathematically, at each step i, the training and test sets are defined as:

$$\text{Train set:} \quad t \in [i, i + T_{\text{train}} - 1] \tag{13}$$

$$\text{Test set:} \quad t \in [i + T_{\text{train}}, i + T_{\text{train}} + T_{\text{test}} - 1] \tag{14}$$

where i represents the starting year of the current window. The window is then rolled forward as:

$$i \leftarrow i + 1 \tag{15}$$

This methodology effectively captures the dynamic nature of market conditions, ensuring that the model is evaluated on previously unseen data in a setting that more accurately reflects the real-world operational environment. Furthermore, this technique enhances the model's generalizability by mitigating the risk of overfitting and providing a more reliable measure of its performance on future data.

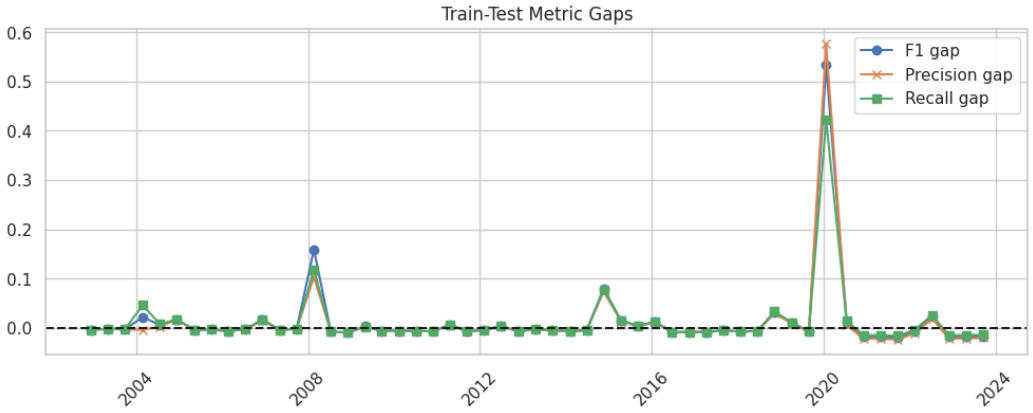
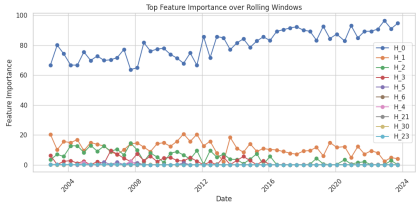


Figure 10. Train-test gap analysis

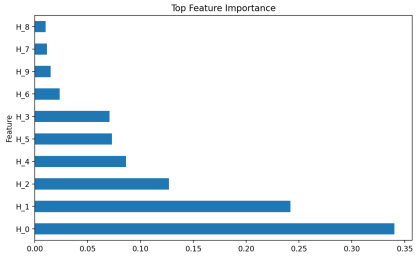
As a result, the model's predictive stability weakened, and the Hurst exponent temporarily lost explanatory power for regime classification. This highlights the need for robustness checks — including stress testing under crisis-like conditions — and supports the use of adaptive learning or rolling recalibration schemes to preserve performance during periods of structural market change.

The rolling window validation analysis (Figure 10) eveals a notable degradation in model performance during specific historical intervals, particularly around the 2008 financial crisis and the 2020 COVID-19 pandemic. This is evidenced by pronounced spikes in the train-test metric gaps—especially in F1-score and recall—indicating a significant drop in generalization capability during these periods. A closer inspection of the temporal evolution of feature importance (Figure 11a) suggests that these performance drops are closely tied to abrupt shifts in the predictive power of the Hurst exponent at short lags (notably H_0 and H_1). During turbulent market phases, the statistical properties of the underlying time

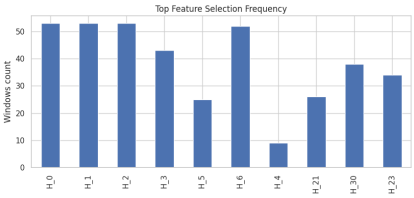
series—such as persistence and long-range dependence—can change rapidly, leading to a mismatch between the training and test distributions within each rolling window. Consequently, the model, which heavily relies on the most recent Hurst values (Figure 11c), may overfit to transient patterns in the training window that fail to generalize to the subsequent test period (Figure 12). This phenomenon underscores the importance of incorporating mechanisms for model adaptation or recalibration in the presence of structural breaks, as well as the potential benefit of integrating additional features that capture regime shifts or exogenous shocks more robustly.



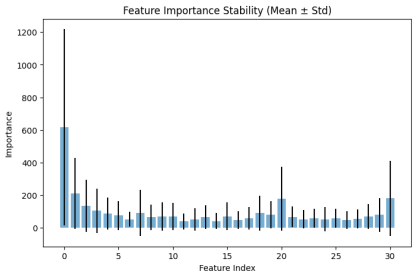
(a) Top features importance over windows rolling.



(b) Mean - σ features importance



(c) Top Frequency features.



(d) Features importance.

Figure 11. Features importance analysis.

To address potential concerns regarding overfitting and the validity of the high reported performance, we conducted a comprehensive learning curve and sensitivity analysis of the model. The optimized LightGBM model achieves very high training accuracy (>95%), but test set performance exhibits variability across rolling windows, particularly for the High Volatility class, highlighting clear overfitting in some intervals. Low and Moderate volatility regimes are predicted relatively reliably, while rare high-volatility periods remain more challenging to classify, requiring caution when interpreting signals under extreme conditions. Feature importance stability tests (Figure 11c) indicate that the most recent Hurst lags (H_0–H_2) primarily drive predictions, although H_0 shows significant variability across windows, especially during crises (e.g., 2008 and 2020), suggesting that the model adapts its dependence on features in response to particularly turbulent market dynamics. Validation of performance across different volatility regimes confirms that, while the model retains some predictive capability for the more common regimes, overfitting and poor generalization under extreme conditions significantly limit its overall reliability. These results emphasize the need for additional methodological interventions, such as stronger regularization, parsimonious feature selection, and rolling recalibration, to mitigate overfitting and improve model robustness.

The results of the feature importance analysis(Figure 11b) clearly highlight the predominance of the shortest lags of the Hurst exponent (particularly H_0, H_1, and H_2) in the classification of volatility regimes. This evidence supports the hypothesis that market persistence, when measured in close temporal proximity to the current observation, serves as a highly informative indicator for distinguishing between low, medium, and high volatility phases. The systematic decline in importance as the lag increases suggests that the relevant memory of the underlying stochastic process is confined to a short temporal horizon, consistent with the notion of markets driven by short-term dynamics.

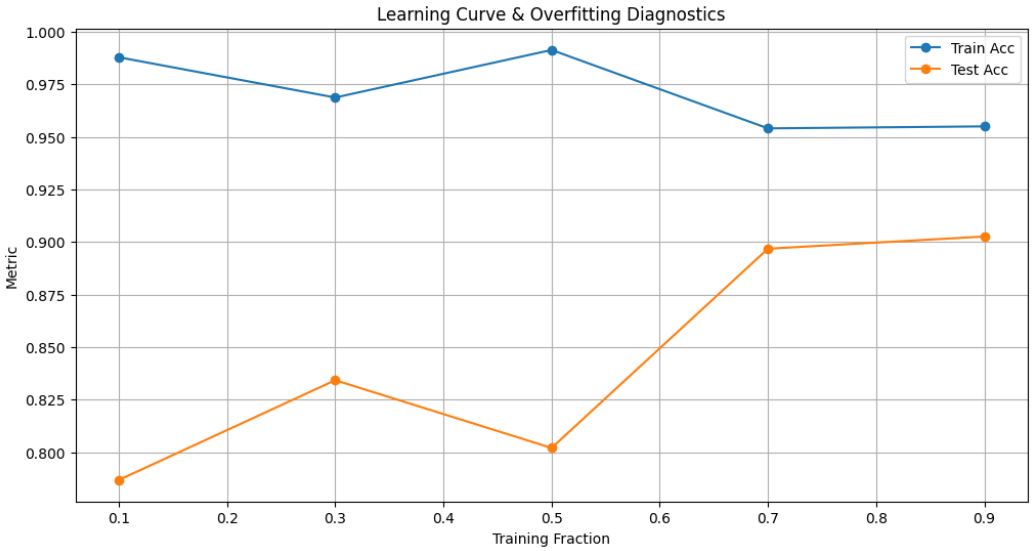


Figure 12. Learning curve analysis.

Moreover, the temporal analysis of feature importance across rolling windows from 2004 to 2024 reveals that the relative importance of H_0 increased significantly during periods of market turbulence (e.g., the 2008 financial crisis, the 2014 oil price collapse, and the 2020 COVID-19 pandemic). This indicates that the model tends to rely more heavily on the most recent Hurst signals in the presence of exogenous shocks. Conversely, during relatively stable periods, importance is more evenly distributed across intermediate lags, suggesting a greater relevance of the market's memory structure.

These findings carry significant methodological and operational implications. From a modeling perspective, they support the adoption of parsimonious feature selection strategies, limiting the input to a small set of highly informative lags (e.g.,H_0, H_3), with benefits in terms of model interpretability and robustness.

Figure 13 illustrates the sensitivity of the optimized LightGBM model to the number of selected features. The training F1-score remains consistently high across all feature subsets, while the validation F1-score stabilizes around 0.96 with minor fluctuations. This indicates that the model is highly expressive and capable of fitting the training data even with a reduced number of features. However, adding too many features does not lead to further improvements in validation performance and may even introduce noise or redundancy—raising an overfitting alert. This behavior is consistent with the feature importance analysis, which showed that the shortest lags of the Hurst exponent (H_0, H_1, H_2) are the most informative, while longer lags contribute only marginally. This reinforces the value of a parsimonious feature selection strategy to enhance generalization and model robustness.

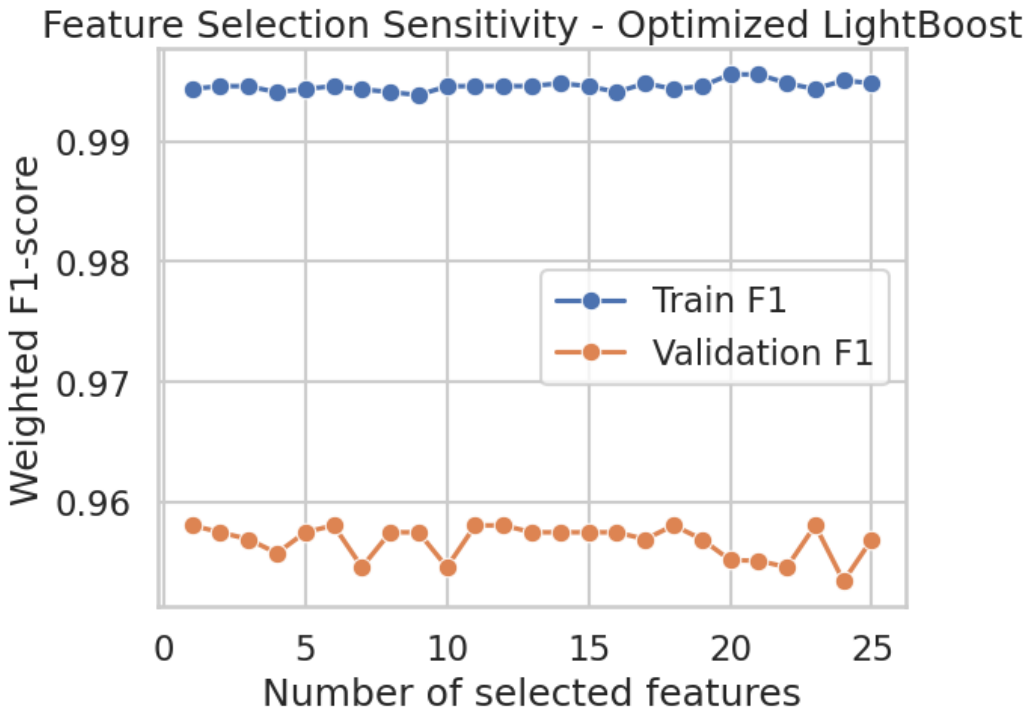


Figure 13. Lgbt optimization.

To further investigate whether the observed performance degradation during specific periods (Figure 14) reflects isolated anomalies or systematic structural limitations, we conducted an event-based diagnostic analysis focused on major market disruptions. In particular, we explicitly examined the following episodes: Hurricane Katrina (2005), the Subprime Financial Crisis (2008–2009), the COVID-19 pandemic (2020), and the Ukraine War (2022), alongside benchmark pre-crisis and post-crisis normal regimes.

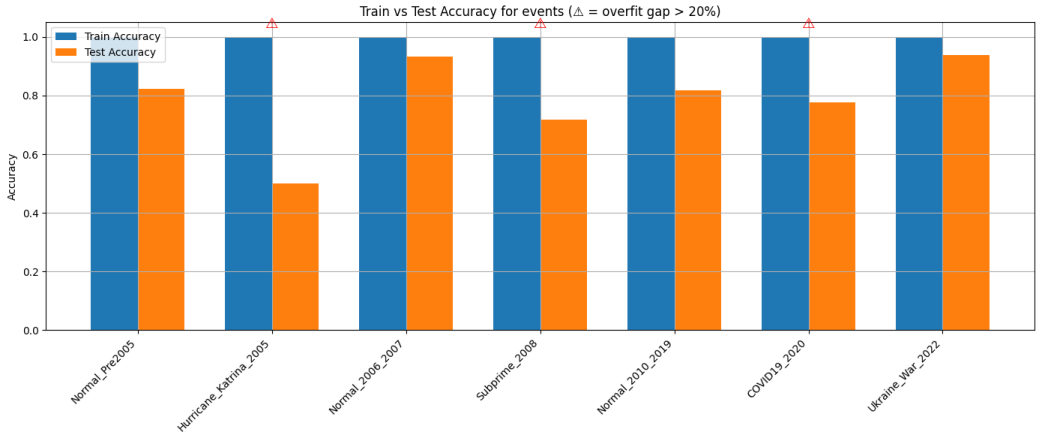


Figure 14. Overgap train-test accuracy analysis.

For each event window, we evaluated (i) the train–test accuracy gap as an explicit measure of overfitting (Figure 14), (ii) the stability of feature importance profiles (Figure 15), and (iii) the concentration

of feature usage through entropy-based measures. Overfitting was flagged when the train–test accuracy gap exceeded a 20% threshold, indicating a substantial loss of generalization. The results reveal that all major crisis periods exhibit pronounced overfitting gaps, in contrast to more stable market regimes, suggesting that the model systematically struggles under extreme and rapidly evolving conditions.

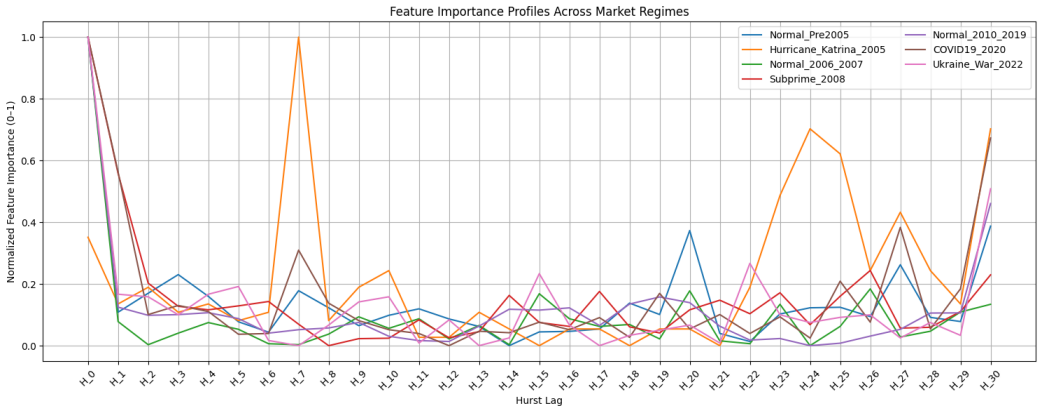


Figure 15. Normalized features important for crisis event.

Feature importance entropy provides additional insight into the model's behavior across regimes (table 7). High entropy values observed during stable periods (e.g., Normal_Pre2005, Normal_2010–2019) indicate a relatively balanced reliance across multiple Hurst lags, consistent with a stable memory structure. In contrast, crisis periods such as Hurricane Katrina and the Subprime Crisis are characterized by lower entropy, reflecting a sharp concentration of importance on a limited subset of features, predominantly the most recent Hurst lags. This collapse in effective feature diversity signals increased model fragility and a heightened risk of overfitting during structural breaks.

Table 7. Entropy of normalized feature importance distributions across different market regimes. Lower entropy indicates stronger concentration on a limited subset of features.

Event	Feature Importance Entropy
Normal_Pre2005	2.988
Hurricane_Katrina_2005	2.913
Normal_2006_2007	2.648
Subprime_2008	2.957
Normal_2010_2019	2.799
COVID19_2020	2.835
Ukraine_War_2022	2.776

To formally assess whether these effects are driven by distributional shifts in the input space, we performed Kolmogorov-Smirnov tests on the dominant feature (H_0), using the pre-2005 period as a reference distribution (table 8). The results show negligible distributional differences during the baseline regime, but statistically significant shifts for all crisis events, with particularly large KS statistics during Hurricane Katrina, the Subprime Crisis, COVID-19, and the Ukraine War. These findings confirm the presence of severe covariate shift during turbulent periods, forcing the model to operate outside the support of the training distribution.

Table 8. Kolmogorov–Smirnov test results for the most influential feature (H_0), using the pre-2005 period as reference. Low p-values indicate statistically significant distributional shifts.

Event	KS Statistic	p-value
Normal_Pre2005	0.017	9.96×10^{-1}
Hurricane_Katrina_2005	0.457	1.05×10^{-18}
Normal_2006_2007	0.472	4.44×10^{-68}
Subprime_2008	0.385	6.83×10^{-37}
Normal_2010_2019	0.463	1.10×10^{-139}
COVID19_2020	0.273	1.82×10^{-12}
Ukraine_War_2022	0.603	3.05×10^{-61}

Taken together, these analyses demonstrate that the performance drops observed during Hurricane Katrina are not isolated incidents, but part of a broader and systematic pattern affecting all major crisis periods. While the model exhibits strong predictive capability under stationary or moderately evolving conditions, its reliability deteriorates sharply in the presence of abrupt structural breaks. This highlights intrinsic limitations of the proposed framework in extreme regimes and underscores the necessity of adaptive strategies—such as rolling retraining, regime-aware regularization, or explicit change-point detection—to ensure robustness in practical deployment.

To further enhance the reliability of the predicted probabilities, following (Manokhin, 2022), isotonic regression was applied as a post-processing step for the calibration of the probability.

$$\hat{p}_i = \hat{g}(x_i)$$

where:

- $\hat{p}_i$ represents the predicted probability with a hat above it, indicating an estimate.
- $\hat{g}(x_i)$ represents the calibration function applied to the input x_i.

Isotonic regression is particularly useful when the model tends to overestimate or underestimate probabilities, ensuring that the predicted probabilities remain monotonic with respect to the true likelihood of events.

This calibration technique was used after model training to adjust the predicted probabilities, aligning them more closely with the actual observed frequencies. By imposing a non-decreasing constraint on the predicted probabilities, isotonic regression helps ensure that higher predicted probabilities correspond to more likely events.

For probability calibration, isotonic regression seeks to minimize the squared error between the observed probabilities and the predicted probabilities while maintaining a monotonic calibration function. Formally, the optimization can be formulated as follows:

$$\min_{\hat{g}(\cdot)} \sum_{i=1}^{n} (y_i - \hat{g}(\hat{p}_i))^2$$

subject to:

$$\hat{g}(\hat{p}_1) \leq \hat{g}(\hat{p}_2) \leq \cdots \leq \hat{g}(\hat{p}_n \quad \text{for} \quad \hat{p}_1 \leq \hat{p}_2 \leq \cdots \leq \hat{p}_n$$

The classification model demonstrates notable performance as evidenced by an overall accuracy of 91% after the hyper-parameters fine-tuning and calibration procedure. This high accuracy indicates that the model effectively differentiates between the various market phases, which highlights its utility in the context of financial analysis.

Turning to the High-Volatility phase, the model performance is particularly good. It records a perfect recall of 1.00, which indicates that the model correctly identified all actual instances classified as high volatility.

This is a critical achievement, especially in financial markets, where the timely identification of crises can be crucial for risk management and decision-making. However, the precision for this phase is somewhat lower at 0.80, indicating that a portion of the instances predicted as high volatility may belong to other classes.

Nevertheless, the F1 score of 0.89 for this category indicates a balanced performance, underscoring the model's effectiveness in distinguishing genuine crisis situations while acknowledging the slight trade-off in precision.

These results are presented in Tables 9 and 10.

Table 9. Lightboost performance on trainset

	Precision	Recall	F1-Score	Support
Moderate volatility	0.92	0.95	0.93	1294
High volatility	0.94	1.00	0.95	217
Low volatility	0.94	0.95	0.93	3111
Accuracy			0.92	4622

Table 10. Lightboost performance results on testset after calibration.

	Precision	Recall	F1-Score	Support
Moderate volatility	0.89	0.93	0.90	539
High volatility	0.90	1.00	0.92	63
Low volatility	0.95	0.85	0.90	554
Accuracy			0.91	1156

The findings of this study are consistent with prior theoretical research on the relationship between market volatility and the Hurst exponent. Specifically, the developed model confirms the results of Bianchi and Mattera (2022) and Cho and Lee (2022), which suggest a meaningful link between both realized and implied volatility and long-range dependence as captured by the Hurst exponent. Moreover, this work offers a novel perspective on the identification of volatility regimes by proposing a clustering approach grounded in a theoretical framework of market inefficiencies. Unlike conventional volatility segmentation methods, which often rely solely on statistical features or historical patterns, the approach presented here integrates a theoretical construct—persistent deviations from market efficiency—to guide the clustering process. This not only strengthens the interpretability of the identified regimes but also bridges the gap between theory and empirical segmentation in financial time series.

7. Conclusions

This study provides a comprehensive analysis of crude oil price fluctuations from January 1, 2000, to January 1, 2024, with a particular focus on applying statistical and machine learning techniques for market phase classification. Beyond merely classifying volatility regimes, the study frames the Hurst exponent as a dynamic measure of market efficiency. Shifts in the Hurst exponent are interpreted as indicative of temporary inefficiencies, offering a richer understanding of market dynamics than conventional volatility metrics alone.

A key aim was to explore the relationship between crude oil price volatility and the Hurst exponent to forecast distinct market conditions, including Low-Volatility, Moderate-Volatility, and High-Volatility phases. A K-means clustering model segmented the dataset into volatility regimes, facilitating deeper insight into market dynamics over time. Descriptive analysis of crude oil prices emphasized fundamental characteristics such as mean, standard deviation, and the range of key variables, including adjusted prices, realized volatility, the Hurst exponent, and log returns. Stationarity was verified via the Augmented Dickey-Fuller (ADF) test, ensuring the reliability of subsequent modeling.

Clustering identified three primary volatility regimes with a silhouette score of 0.60, indicating well-separated market states. These clusters served as the foundation for supervised classification, where the Hurst exponent was a core feature. LightGBM achieved the highest predictive performance with 88% accuracy, particularly excelling in identifying transition and stable-trend phases. Hyper-parameter tuning through Optuna and Bayesian optimization further improved accuracy to 89

To address overfitting and extreme market events, a detailed analysis across rolling windows was conducted. Training accuracy remained high (¿95%), while test accuracy exhibited variability, especially for the High-Volatility class, revealing overfitting during crisis windows such as Hurricane Katrina (2005), the Subprime crisis (2008–2009), and COVID-19 (2020). Low- and Moderate-Volatility regimes were predicted more reliably. Feature importance stability tests highlighted that the shortest Hurst lags (H_0–H_2) were most influential, although H_0 showed high variability during crises, reflecting adaptive reliance on recent market information.

Overfitting was formally quantified using the train-test accuracy gap, with gaps exceeding 20% during turbulent periods. Entropy analysis of feature importance indicated moderate to high concentration of predictive power in early Hurst lags. Higher entropy reflected a more uniform distribution of importance across features, while lower entropy indicated concentration on specific lags. This confirms that the model relies heavily on recent Hurst measures during crises, whereas in more stable periods, importance spreads more evenly across intermediate lags. Kolmogorov-Smirnov tests further confirmed structural shifts in feature distributions during crises, explaining observed overfitting and performance variability.

Importantly, the crude oil market exhibits unique structural characteristics that must be considered when interpreting these results. Price dynamics are strongly influenced by OPEC+ production decisions, geopolitical tensions, supply chain disruptions, and inventory cycles, generating nonlinear and episodic volatility bursts. Pronounced seasonality—driven by refining cycles and global consumption patterns—introduces additional time-varying components in volatility and long-memory dynamics. Furthermore, the deep and liquid futures market, with frequent contract roll-overs, can create mechanical patterns in realized volatility not observed in spot-dominated markets. These factors contribute to the observed reliance on recent Hurst lags and to periods of extreme volatility where predictive performance declines.

Consequently, caution is warranted when generalizing the methodology to other assets. Equity, foreign exchange, fixed-income, and cryptocurrency markets exhibit distinct microstructural features, liquidity profiles, and systemic drivers, meaning the Hurst–volatility relationship may not hold universally. Researchers aiming to apply this framework elsewhere should adjust Hurst estimation windows, recalibrate clustering stages, consider additional macroeconomic or liquidity indicators, and perform stress testing across historically turbulent periods.

Overall, while the proposed methodology demonstrates strong predictive power for crude oil, predictive reliability declines during structural breaks and extreme events. Methodological interventions such as stronger regularization, parsimonious feature selection, and rolling recalibration are recommended to enhance robustness. Future work should also explore extending the approach to other asset classes, incorporating real-time forecasting, and integrating additional features to account for asset-specific drivers of volatility. These extensions could broaden the applicability of the approach, while maintaining caution regarding the limitations imposed by market-specific microstructures.

Author contributions

All authors equally contributed to this study.

Funding

Laura Molero González is funded by the Ministerio de Ciencia, Innovación y Universidades of Spain (FPU2023).

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

Ahn JJ, Oh KJ, Kim TY, et al. (2011) Usefulness of support vector machine to develop an early warning system for financial crisis. *Expert Syst Appl* 38: 2966–2973. https://doi.org/10.1016/j.eswa.2010.08.085

Akiba T, Sano S, Yanase T, et al. (2019) Optuna: A next-generation hyperparameter optimization framework. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–2631. https://doi.org/10.1145/3292500.333070

Alvarez-Ramirez J, Cisneros M, Ibarra-Valdez C, et al. (2002) Multifractal hurst analysis of crude oil prices. *Physica A* 313: 651–670. https://doi.org/10.1016/S0378-4371(02)00985-8

Angelini D, Bianchi S (2023) Nonlinear biases in the roughness of a fractional stochastic regularity model. *Chaos Soliton Fract* 172: 113550. https://doi.org/10.1016/j.chaos.2023.113550

Bašta M, Molnár P (2018) Oil market volatility and stock market volatility. *Financ Res Lett* 26: 204–214. https://doi.org/10.1016/j.frl.2018.02.001

Bianchi S, Di Sciorio F, Mattera R (2022) Forecasting vix with hurst exponent In: *Methods and Applications in Fluorescence. Cham: Springer International Publishing*, Springer International Publishing, 90-95. https://doi.org/10.1007/978-3-030-99638-3_15

Bianchi S (2005) Pathwise identification of the memory function of multifractional brownian motion with application to finance. *Int J Theor Appl Financ* 8: 255–281. https://doi.org/10.1142/S0219024905002937

Bianchi S, Pianese A (2014) Asset price modeling: From fractional to multifractional processes. *Future perspect Risk Models Finance* 18: 247–285. https://doi.org/10.1007/978-3-319-07524-2_7

Carmona P, Coutin L, Montseny G (2003) Stochastic integration with respect to fractional brownian motion. *Ann Inst Henri Poincare-Probab Stat* 39: 27–68. https://doi.org/10.1016/S0246-0203(02)01111-1

Chatzis SP, Siakoulis V, Petropoulos A, et al. (2018) Forecasting stock market crisis events using deep and statistical machine learning techniques. *Expert Syst Appl* 112: 353–371. https://doi.org/10.1016/j.eswa.2018.06.032

Chen Y, Härdle WK, Pigorsch U (2010) Localized realized volatility modeling. *J Am Stat Assoc* 105: 1376–1393. https://doi.org/10.1198/jasa.2010.ap09039

Cho P, Lee M (2022) Forecasting the volatility of the stock index with deep learning using asymmetric hurst exponents. *Fractal Fract* 6: 394. https://doi.org/10.3390/fractalfract6070394

Di Sciorio F (2023) Clustering analysis on hurst dynamic. *Stud App Econ*. https://doi.org/10.25115/sae.v42i1.9527

Di Sciorio F, Mattera R, Segovia JET (2023) Measuring conditional correlation between financial markets' inefficiency. *Quant Financ Econ* 7: 491–507. https://doi.org/10.3934/QFE.2023025

Fernández-Martínez M, Sánchez-Granero MA, Muñoz Torrecillas MJ, et al. (2017) A comparison of three hurst exponent approaches to predict nascent bubbles in sp500 stocks. *Fractals* 25. https://doi.org/10.1142/S0218348X17500062

Granero MAS, Segovia JET, Pérez JG (2008) Some comments on hurst exponent and the long memory processes on capital markets. *Physica A* 387: 5543–5551. https://doi.org/10.1016/j.physa.2008.05.053

Kristoufek L, Vosvrda M (2014a) Commodity futures and market efficiency. *Energ Econ* 42: 50–57. https://doi.org/10.1016/j.eneco.2013.12.001

Kristoufek L, Vosvrda M (2014b) Measuring capital market efficiency: long-term memory, fractal dimension and approximate entropy. *Eur Phys J B* 87: 1–9. https://doi.org/10.1140/epjb/e2014-50113-6

Kristoufek L, Vosvrda M (2016) Gold, currencies and market efficiency. *Physica A* 449: 27–34. https://doi.org/10.1016/j.physa.2015.12.075

Lin WY, Hu YH, Tsai CF (2011) Machine learning in financial crisis prediction: a survey. *IEEE Ieee T Syst Man Cy C* 42: 421–436.

Mandelbrot BB (1982) The many faces of scaling: fractals, geometry of nature, and economics. In: *Self-Organization and Dissipative Structures: Applications in the Physical and Social Sciences*, 91–109. University of Texas Press.

Mandelbrot BB, Wallis JR (1969) Robustness of the rescaled range r/s in the measurement of noncyclic long run statistical dependence. *Water Resour Res* 5: 967–988. https://doi.org/10.1029/WR005i005p00967

Manokhin V (2022) *Machine learning for probabilistic prediction*. Phd thesis, Royal Holloway, University of London.

Mattera R, Di Sciorio F, Trinidad-Segovia JE (2022) A composite index for measuring stock market inefficiency. *Complexity* 1: 9838850. https://doi.org/10.1155/2022/9838850

Mensi W, Beljid M, Managi S (2014) Structural breaks and the time-varying levels of weak-form efficiency in crude oil markets: Evidence from the hurst exponent and shannon entropy methods. *Int Econ* 140: 89–106. https://doi.org/10.1016/j.inteco.2014.10.001

Peltier RF, Levy-Vehel J (1995) Multifractional brownian motion: Definition and preliminary results. *Inria Res Rep*.

Samitas A, Kampouris E, Kenourgios D (2020) Machine learning as an early warning system to predict financial crisis. *Int Rev Financ Anal* 71: 101507. https://doi.org/10.1016/j.irfa.2020.101507

Yao CZ, Liu C, Ju WJ (2020) Multifractal analysis of the wti crude oil market, us stock market and epu. *Physica A* 550: 124096. https://doi.org/10.1016/j.physa.2019.124096

A. Supplementary Material: AMBE Implementation and Sensitivity Analysis

This Supplementary Material provides Python code for:

1. Estimating the Hurst exponent using the AMBE, R/S, and variance methods.
2. Conducting a full sensitivity analysis of AMBE parameters (δ, q, k) under different market regimes.

```python
import numpy as np
import matplotlib.pyplot as plt
import pywt
import pandas as pd
from scipy.special import gamma
from fbm import fbm

# ------------------------
# 1. AMBE (Absolute Moment Based Estimator) method
# ------------------------
def AMBE_theoretical(X, delta=30, q=1, k=2):
    """
    Compute the Hurst exponent of a time series using the AMBE method.
```

```
14
15      Parameters:
16      X     : array-like, input time series
17      delta : int, rolling window size
18      q     : int, lag for increments
19      k     : int, order of the absolute moment
20
21      Returns:
22      mean Hurst exponent over the series
23      """
24      X = np.asarray(X)  # Ensure X is a NumPy array
25      n = len(X)         # Length of the time series
26      hurst_estimates = np.zeros(n - delta)  # Array to store rolling estimates
27      K_k = 2**(k/2) * gamma((k+1)/2) / np.sqrt(np.pi)  # Normalization constant
28
29      # Loop over the rolling window
30      for i in range(delta, n):
31          window = X[i - delta:i + 1]  # Current rolling window
32          # Compute k-th absolute moment of increments
33          S_k = np.mean(np.abs(window[q:] - window[:-q])**k)
34          numerator = np.log(np.sqrt(np.pi) * S_k / (2**(k/2) * gamma((k+1)/2) *
                K_k))
35          denominator = k * np.log((n + 1)/q)
36          H = - numerator / denominator  # AMBE Hurst estimate
37          hurst_estimates[i - delta] = H
38
39      return np.mean(hurst_estimates)  # Return the mean over rolling estimates
40
41  # ------------------------
42  # 2. R/S analysis method
43  # ------------------------
44  def hurst_RS(X):
45      """
46      Estimate the Hurst exponent using the Rescaled Range (R/S) method.
47
48      Parameters:
49      X : array-like, input time series
50
51      Returns:
52      Hurst exponent estimate
53      """
54      X = np.asarray(X)
55      N = len(X)
```

```
    max_k = N // 2
    R_S = []
    n_vals = []

    # Loop over different window sizes to compute rescaled range
    for n in range(10, max_k, max_k // 20):
        m = N // n  # Number of segments
        R_S_vals = []
        for i in range(m):
            segment = X[i*n:(i+1)*n]
            Y = np.cumsum(segment - np.mean(segment))  # Cumulative deviation
                from mean
            R = np.max(Y) - np.min(Y)                  # Range
            S = np.std(segment)                        # Standard deviation
            if S > 0:
                R_S_vals.append(R/S)
        if R_S_vals:
            R_S.append(np.mean(R_S_vals))
            n_vals.append(n)

    # Linear regression on log-log plot of R/S vs n to estimate H
    H, _ = np.polyfit(np.log(n_vals), np.log(R_S), 1)
    return H

# ------------------------
# 3. Variance method
# ------------------------
def hurst_variance(X):
    """
    Estimate the Hurst exponent based on the variance of increments.

    Parameters:
    X : array-like, input time series

    Returns:
    Hurst exponent estimate
    """
    X = np.asarray(X)
    N = len(X)
    lags = range(2, N//10)  # Range of lags to consider
    tau = [np.std(np.subtract(X[l:], X[:-l])) for l in lags]  # Std of increments
    H, _ = np.polyfit(np.log(lags), np.log(tau), 1)           # Regression on log
        -log
```

```
    return H

# -----------------------
# 4. Synthetic fractional Brownian motion (fBm) series generator
# -----------------------
def generate_series(N, H):
    """
    Generate a synthetic fBm time series.

    Parameters:
    N : int, length of the series
    H : float, true Hurst exponent

    Returns:
    fBm series of length N
    """
    return fbm(N, H, length=1, method='daviesharte')

# -----------------------
# 5. Test different Hurst estimators on synthetic series
# -----------------------
def test_estimators(N=5000, true_Hs=[0.2, 0.5, 0.7, 0.9], n_sim=100):
    """
    Evaluate AMBE, R/S, and variance methods on synthetic fBm series.

    Parameters:
    N       : int, length of each time series
    true_Hs : list of true Hurst exponents to test
    n_sim   : int, number of simulations per H value

    Returns:
    Dictionary containing estimates for each method and true H
    """
    results = {h: {"AMBE": [], "R/S": [], "Variance": []} for h in true_Hs}

    for true_H in true_Hs:
        for _ in range(n_sim):
            X = generate_series(N, true_H)  # Generate synthetic series
            # Compute Hurst estimates using all methods
            results[true_H]["AMBE"].append(AMBE_theoretical(X))
            results[true_H]["R/S"].append(hurst_RS(X))
            results[true_H]["Variance"].append(hurst_variance(X))

```

```
    return results

# -----------------------
# 6. Summarize results
# -----------------------
def analyze_results(results):
    """
    Compute mean, median, and standard deviation for all estimates.

    Parameters:
    results : dict, output of test_estimators()

    Returns:
    pandas DataFrame with summary statistics
    """
    summary_stats = []
    for h in results:
        for method in results[h]:
            arr = np.array(results[h][method])
            summary_stats.append({
                "True H": h,
                "Method": method,
                "Mean": np.mean(arr),
                "Median": np.median(arr),
                "Std": np.std(arr)
            })
    return pd.DataFrame(summary_stats)

# -----------------------
# 7. Plot results
# -----------------------
def plot_results(results, true_Hs):
    """
    Create boxplots comparing estimated Hurst exponents for each true H.

    Parameters:
    results  : dict, output of test_estimators()
    true_Hs  : list of true H values tested
    """
    fig, axes = plt.subplots(1, len(true_Hs), figsize=(15, 5), sharey=True)
    for i, h in enumerate(true_Hs):
        data = [results[h][m] for m in ["AMBE", "R/S", "Variance"]]
        axes[i].boxplot(data, labels=["AMBE", "R/S", "Variance"])
```

```
        axes[i].set_title(f"True H = {h}")
        axes[i].axhline(y=h, color="red", linestyle="--", label="True H")
        axes[i].legend()
    plt.suptitle("Comparison of Hurst Estimators")
    plt.show()

# -----------------------
# 8. Main execution
# -----------------------
if __name__ == "__main__":
    np.random.seed(42)  # For reproducibility
    true_Hs = [0.2, 0.5, 0.7, 0.9]

    # Run the simulations
    results = test_estimators(N=1000, true_Hs=true_Hs, n_sim=1_000)

    # Summarize and display the results
    df_summary = analyze_results(results)
    print(df_summary)

    # Plot comparison of methods
    plot_results(results, true_Hs)
```

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.special import gamma
from fbm import fbm
import seaborn as sns
import itertools

# -----------------------
# 1. Fractional Brownian Motion (fBm) series generator
# -----------------------
def generate_fbm(N, H):
    """
    Generate a fractional Brownian motion (fBm) time series.

    Parameters:
    N : int, length of the series
    H : float, true Hurst exponent

    Returns:
    fBm series of length N
```

```
    """
    return fbm(N, H, length=1, method='daviesharte')

# ------------------------
# 2. Full sensitivity analysis of AMBE parameters
# ------------------------
def sensitivity_full(N=500, H_values=[0.3,0.5,0.7], deltas=[10,20,30], qs=[1,2,5], ks=[1,2,3], n_sim=50):
    """
    Evaluate the impact of AMBE parameters (delta, q, k) on the estimated Hurst exponent.

    Parameters:
    N        : int, length of each fBm series
    H_values : list of true Hurst exponents to test
    deltas   : list of rolling window sizes
    qs       : list of lags for increments
    ks       : list of moment orders
    n_sim    : int, number of simulations per parameter combination

    Returns:
    results : list of dictionaries containing mean and std of H estimates
    """
    results = []

    for H_true in H_values:
        print(f"Simulating for H={H_true}")  # Progress indicator
        # Iterate over all combinations of delta, q, k
        for delta, q, k in itertools.product(deltas, qs, ks):
            H_est_list = []
            # Run multiple simulations for statistical robustness
            for _ in range(n_sim):
                X = generate_fbm(N, H_true)                    # Generate synthetic fBm series
                H_est = AMBE_theoretical(X, delta=delta, q=q, k=k)  # Estimate H using AMBE
                H_est_list.append(H_est)
            # Store summary statistics for this parameter combination
            results.append({
                "H_true": H_true,
                "delta": delta,
                "q": q,
                "k": k,
```

```
                "H_mean": np.mean(H_est_list),
                "H_std": np.std(H_est_list)
            })
    return results

# -----------------------
# 3. Heatmap visualization of sensitivity results
# -----------------------
def plot_heatmaps(results, H_true_value):
    """
    Plot heatmaps showing mean and standard deviation of H estimates
    for different AMBE parameter combinations, for a given true H.

    Parameters:
    results      : list of dictionaries from sensitivity_full()
    H_true_value : float, the true H value to filter results
    """
    # Filter results for the specified true H
    data = [r for r in results if r['H_true']==H_true_value]

    # Extract unique delta values and q,k combinations
    deltas = sorted(set(r['delta'] for r in data))
    qk = sorted(set((r['q'], r['k']) for r in data))

    # Initialize matrices for mean and std
    mean_matrix = np.zeros((len(deltas), len(qk)))
    std_matrix = np.zeros((len(deltas), len(qk)))

    # Fill matrices with corresponding H_mean and H_std
    for i, delta in enumerate(deltas):
        for j, (q, k) in enumerate(qk):
            row = next(r for r in data if r['delta']==delta and r['q']==q and r['
                k']==k)
            mean_matrix[i, j] = row['H_mean']
            std_matrix[i, j] = row['H_std']

    # Create labels for x-axis
    labels = [f"q={q},k={k}" for (q,k) in qk]

    # Plot heatmap for mean H estimates
    plt.figure(figsize=(15,6))
    sns.heatmap(mean_matrix, annot=True, xticklabels=labels, yticklabels=deltas,
        cmap="viridis")
```

```
    plt.title(f"Mean H estimate for H_true={H_true_value}")
    plt.xlabel("q,k combinations")
    plt.ylabel("delta")
    plt.show()

    # Plot heatmap for standard deviation of H estimates
    plt.figure(figsize=(15,6))
    sns.heatmap(std_matrix, annot=True, xticklabels=labels, yticklabels=deltas,
        cmap="magma")
    plt.title(f"Std of H estimate for H_true={H_true_value}")
    plt.xlabel("q,k combinations")
    plt.ylabel("delta")
    plt.show()

# ------------------------
# 4. Main execution for sensitivity analysis
# ------------------------
if __name__ == "__main__":
    np.random.seed(42)  # Ensure reproducibility

    # Parameters for simulation
    N = 500
    H_values = [0.3, 0.5, 0.7]          # Different market regimes
    deltas = [10, 20, 30]               # Rolling window sizes
    qs = [1, 2, 5]                       # Lags for increments
    ks = [1, 2, 3]                       # Orders of absolute moments
    n_sim = 50                            # Number of simulations per combination

    # Run full sensitivity analysis
    results = sensitivity_full(N=N, H_values=H_values, deltas=deltas, qs=qs, ks=
        ks, n_sim=n_sim)

    # Plot heatmaps for each H_true
    for H_true in H_values:
        plot_heatmaps(results, H_true)
```