Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometrics & Operations Research

Advancing Cryptocurrency Volatility Forecasting With Local Linear Forests

Joep Noot (619181)

Supervisor:	Tetereva A.
Second assessor:	Mueller M.
Date:	1st July 2024

Abstract

Amid the rapid evolution of financial markets, digital currencies have emerged, marked by their high volatility, innovation and transforming traditional investment approaches. This study addresses the need for reliable volatility forecasting tools in cryptocurrency markets and investigates the effectiveness of Local Linear Forests, an innovative approach by Friedberg et al. (2020), that adapts to local data structures, making it well-suited for the dynamic nature of cryptocurrency markets. Its performance is compared to established benchmarks like GARCH, GJR-GARCH, HAR-RV, and Random Forests. Our findings indicate that LLFs consistently outperform these models in out-of-sample volatility forecasting and show superiority in yielding utility benefits. This significance in superiority is shown through Model Confidence Sets. Furthermore, significant differences in forecasting errors are observed between different phases of market cycles. Additionally, volatility forecasts for mid-cap currencies exhibit relatively lower forecast errors compared to large-cap currencies. The source code for the analyses can be found on GitHub.

1 Introduction

In today's rapidly evolving financial landscape, digital currencies emerged as a disruptive force, characterized by their erratic price movements and revolutionary effect on conventional investment paradigms. In contrast to conventional financial markets that are driven by economic indicators or geopolitical events, cryptocurrency markets are uniquely influenced by technological advancements, legislative changes, and real-time social media trends. An additional layer of complexity is added because of the growing popularity and adoption of cryptocurrencies. Originally a market dominated by tech enthusiasts, it has now grown to include institutional investors, individual traders, and global financial institutions. The demand and need for reliable and robust forecasting instruments become increasingly critical, as these factors shape the market dynamics.

However, this shift goes beyond cryptocurrency trading and investment. In an era where blockchain technology is reshaping conventional business models (Nowiński and Kozma, 2017) and altering the digital landscape, managing volatility in cryptocurrency markets assumes critical significance. Predicting and controlling volatility in these markets is crucial, as seen by the growing usage of blockchain across various sectors, such as finance, supply chain management (Gurtu and Johny, 2019), healthcare (Agbo et al., 2019), and governance (Lumineau et al., 2021). Accurate forecasting is essential for reducing financial risks as well as guaranteeing the reliability and security of blockchain applications across these diverse fields.

These factors create a complex environment where traditional forecasting methods - and even the more modern tools - often fall short of capturing the many nuances of this dynamic world (Murray et al., 2023). Therefore, this study explores the potential of Local Linear Forests (LLFs), an innovative approach introduced by Friedberg et al. (2020), to forecast volatility within cryptocurrency markets. One of the key strengths of LLFs is their ability to adapt to local structures within the data. Unlike global models that apply a single relationship across the entire dataset, LLFs can fit different linear models to different data regions. This local adaptability allows LLFs to better capture sudden changes and unique patterns characteristic of cryptocurrency markets, such as abrupt price shifts or spikes in volatility.

This study evaluates the effectiveness of LLFs by comparing them to established models like Generalized Autoregressive Conditional Heteroskedasticity (GARCH) (Bollerslev, 1986), GJR-GARCH (Glosten et al., 1993), Heterogeneous Autoregressive model based on Realized Volatility (HAR-RV) (Corsi, 2009) and Random Forest (RF) (Breiman, 2001). This comparison sheds light on the relative performance of these models and their potential advantages in forecasting volatility in the cryptocurrency market. Additionally, the study will explore how these models perform across different phases of the market cycle — including bull runs, bear markets, and consolidation periods — to assess their adaptability to varying market conditions. This analysis is pivotal for enhancing risk management strategies and facilitating informed investment decision-making.

Furthermore, this study evaluates the utility benefits of volatility models for risk-targeting investors by adopting the framework proposed by Bollerslev et al. (2018), which prioritizes mean-variance preferences. This framework allocates wealth optimally by considering both expected returns and risk, measured through volatility. Through empirical analysis, the study assesses the expected utility per unit of wealth based on the volatility forecasts and investigates whether LLFs can provide substantial economic benefits compared to baseline models.

Moreover, as the cryptocurrency market is diverse, encompassing a broad spectrum of digital assets with varying market capitalizations, this study explores whether LLFs demonstrate distinct performance characteristics when forecasting for large-cap cryptocurrencies like Bitcoin and Ethereum compared to mid-cap altcoins such as Litecoin and Polygon. By assessing the robustness and efficacy of LLFs across these different market segments, we aim to offer a comprehensive insight into their suitability and effectiveness within the cryptocurrency market.

This study illustrates that Local Linear Forests excel over baseline models in out-of-sample volatility forecasting. Additionally, LLFs exhibit a slight edge in utility benefits, yielding the highest realized utility across most cryptocurrencies. In our investigation of market phases, we identify substantial variations in forecasting errors between consolidation and bull phases, although LLFs are less affected by this compared to the baseline models. Finally, compared to large-cap currencies, mid-cap currency volatility estimates show substantially lower forecast errors. The source code for the analyses can be found on GitHub.

The paper will proceed as follows. Section 2 provides an in-depth review of the existing literature on volatility forecasting, with a specific focus on cryptocurrencies. Section 3 discusses the data used in this study. Section 4 outlines the baseline models employed in this research, serving as benchmarks against the Local Linear Forest. Section 5 establishes the framework for the study, detailing the LLF model and the methodology for model evaluation. The findings of this evaluation are presented in Section 6. The paper concludes with Section 7.

2 Literature Review

For decades, forecasting has been a focal point of both academic research and practical application in financial markets. A significant driving factor behind this research is the recognition of volatility as a key determinant of investment risk and return. The evolution of forecasting techniques can be traced through significant milestones in financial research and innovation.

Six decades ago, Cootner (1964) proposed that asset prices evolve according to a random walk. According to his theory, past price movements or patterns do not provide any useful information for predicting future price movements. This implies that both volatility and returns would be impossible to predict. However, since this hypothesis, a vast amount of models have been introduced attempting to show the opposite.

Engle (1982) revolutionized volatility modelling with the introduction of the Autoregressive Conditional Heteroskedasticity (ARCH) model, which permits the variance of financial time series to vary over time based on past errors. Building on the ARCH framework, Bollerslev (1986) introduced the Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model. The GARCH model extended the ARCH framework by incorporating lagged volatility terms, thereby enabling more flexible and accurate modelling of volatility dynamics in financial time series.

A variant of the standard GARCH model, the Glosten-Jagannathan-Runkle GARCH (GJR-GARCH) model (Glosten et al., 1993), incorporates asymmetric shocks by allowing different responses to positive and negative past returns or shocks. This flexibility makes it particularly useful for forecasting volatility in financial markets, where asymmetric reactions to market events are common.

Furthermore, the Heterogeneous Autoregressive Realized Volatility (HAR-RV) model, as introduced by Corsi (2009), extends traditional volatility modelling by incorporating realized volatility over multiple time horizons—daily, weekly, and monthly. This structure captures the persistence and long-memory effects of volatility, recognizing that market participants operate and react on different time scales.

Moreover, the application of machine learning techniques in financial forecasting has seen significant evolution in recent years, with a wide variety of models being applied. Among these models is the Random Forest, introduced by Breiman (2001). This ensemble learning technique constructs multiple decision trees and combines their predictions to enhance forecasting accuracy.

The focus of this research, Local Linear Forests, as introduced by Friedberg et al. (2020), extends the Random Forest framework. LLFs are specifically designed to address the challenges posed by nonlinear and dynamic relationships in financial data. Unlike traditional Random Forests, which make predictions based solely on the average of leaf nodes, LLFs incorporate local linear regression models within each node, enhancing their ability to capture complex patterns.

Instead of applying LLFs to the traditional stock market, this research focuses on a less explored market: the cryptocurrency market. Established only fifteen years ago with the introduction of Bitcoin by Nakamoto (2008), the cryptocurrency market has evolved into a unique and rapidly changing landscape, where factors such as technological advancements, regulatory developments, market sentiment, and investor behaviour significantly influence prices and volatility levels. Studies by Kurihara and Fukushima (2017) examined whether cryptocurrency markets, specifically Bitcoin, adhere to the principles of market efficiency (Fama, 1965). Following Bitcoin's inception, researchers and market participants began exploring the distinctive characteristics of cryptocurrencies. For instance, Fink and Johann (2014) demonstrated that Bitcoin exhibits significant price volatility, a non-normal return distribution, and diverse ownership and trading activity, differentiating it from traditional assets. These factors are why cryptocurrencies are perceived as having potential for speculative trading and investment.

To address the complexities of cryptocurrency financial time series, such as extreme observations, asymmetries, and nonlinear characteristics, Catania and Grassi (2017) proposed a dynamic model. Additionally, Kim et al. (2021) employed the Bayesian Stochastic Volatility model and various GARCH models to forecast cryptocurrency volatility. Furthermore, Ma et al. (2020) found statistically significant improvements in forecasting Realized Variance using a Markov regime-switching MIDAS approach.

As the cryptocurrency ecosystem matures and evolves, researchers are increasingly leveraging advanced machine-learning techniques to improve volatility forecasting in cryptocurrencies, given their capability to capture complex patterns and dynamics. Wang et al. (2023) demonstrated that Random Forests and Long Short-Term Memory (LSTM) networks significantly outperform traditional volatility models like GARCH. D'Amato et al. (2022) utilize various neural network architectures for this purpose. Additionally, Liu et al. (2023) employ a range of machine-learning models to forecast cryptocurrency returns. Within the realm of machine-learning methods, one promising approach, Local Linear Forest by Friedberg et al. (2020), has shown effectiveness in realized volatility forecasting (Kleen and Tetereva, 2022). However, its application in forecasting within the cryptocurrency environment remains unexplored and will be the focus of this research.

3 Data

In this study, we will utilize a comprehensive dataset consisting of information from eight distinct cryptocurrencies. This dataset forms the basis for our empirical analysis and model estimation. The data is collected from Crypto Data Download.

3.1 Cryptocurrencies

The dataset is selected to encompass a diverse selection of cryptocurrencies, aiming to provide a representative sample of the cryptocurrency market. Following the approach of Ftiti et al. (2023), this research will focus on four large-cap coins: Bitcoin (BTC), Ethereum (ETH), Tether (USDT), and Binance Coin (BNB). These coins are widely recognized and hold the highest market capitalizations within the cryptocurrency ecosystem.

However, to facilitate deeper exploration beyond the conventional large-cap coins, this dataset includes four mid-cap altcoins. By incorporating a variety of coins, the dataset offers a comprehensive perspective of the cryptocurrency landscape, allowing for a thorough market analysis. These altcoins are selected to encompass the diversity and breadth of the cryptocurrency market beyond the top-ranked coins, with market capitalizations ranging between five and ten billion dollars and data availability from Crypto Data Download. The mid-cap coins included are Bitcoin Cash (BCH), Litecoin (LTC), Internet Computer (ICP), and Polygon (MATIC). Detailed summary statistics for these coins can be found in the Appendix. The dataset covers the period from August 1, 2021, to June 1, 2024.

3.2 Features

Building upon the findings of previous research (Akyildirim et al., 2021; Bâra and Oprea, 2024; Gradojevic et al., 2023), we employ a subset of features and accompanying windows or lags chosen in these studies. These features include a variety of technical indicators, each offering unique insights into market behaviour and volatility dynamics. They are designed to capture different aspects of market activity, including price movements, trading volumes, trend strength, momentum, and market sentiment, among others. The total set of features can be found in the Appendix. Within this feature set is the realized volatility, which is calculated as follows:

$$RV_t = \sqrt{\sum_{j=1}^{K} r_{t-j\cdot\Delta}^2} \tag{1}$$

where $r_{t-j\cdot\Delta}$ are the hourly log-returns of a coin, where $\Delta = 1/K$ and K is equal to twenty-four.

While GARCH models rely solely on historical returns as input data, and Random Forests utilize a set of features, the Local Linear Forests (LLFs) framework introduces an additional layer of complexity. LLFs operate in two distinct phases: the splitting phase, where the feature space is partitioned into smaller regions, and the regression phase, where the model is trained on each partition to predict volatility.

In the regression phase of LLFs, not all variables are included as input features. Traditional financial indicators such as 'open', 'close', 'high', 'low', and 'volume' are often omitted. The rationale behind this exclusion is twofold. Firstly, these basic variables are inherently correlated

with each other and may introduce multicollinearity issues, potentially compromising the stability and interpretability of the model. Secondly, LLFs leverage the hierarchical nature of random forests, enabling interactions between features to be captured in the splitting phase, thereby mitigating the need for explicit inclusion of basic variables in the regression phase.

Moreover, as LLFs employ a regression framework, it becomes imperative to ensure the stationarity of the input features. Non-stationarity in the data can lead to spurious regression results and unreliable forecasts. To address this concern, a preprocessing step is incorporated wherein the stationarity of each feature is assessed using the augmented Dickey-Fuller test (Said and Dickey, 1984). Features exhibiting non-stationarity are transformed by taking first differences, and the stationarity test is repeated iteratively until all features exhibit stationary behaviour. This iterative differencing approach ensures that the model is trained on a stable and statistically sound dataset, enhancing the reliability and accuracy of volatility forecasts.

4 Theoretical Background

This section provides a comprehensive overview of the baseline models used in this research.

4.1 GARCH

As introduced by Bollerslev (1986), the GARCH model allows for a flexible lag structure in modelling volatility. The GARCH model captures volatility clustering, indicating that significant changes in financial time series tend to be followed by additional significant changes. The GARCH model is widely used for modelling and forecasting volatility due to its ability to model time-varying volatility and its relatively simple estimation process. As demonstrated by Engle (2001), the GARCH model has been highly effective in capturing the dynamic nature of financial markets and has become a standard tool in academic research and industry applications.

The model specification for the GARCH(p, q) model can be expressed as follows:

$$\sigma_t^2 = \omega + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2 \tag{2}$$

where

- σ_t^2 is the conditional variance at time t,
- ε_{t-1} is the error term at time $t-1$,
- ω, α, and β are parameters to be estimated. These parameters can be estimated using maximum likelihood estimation.

4.2 GJR-GARCH

The GJR-GARCH model, introduced by Glosten et al. (1993), extends the traditional GARCH model by allowing for asymmetric effects of positive and negative shocks on volatility. This model captures the "leverage effect," where negative shocks increase volatility more than positive shocks of the same magnitude, making it a more realistic representation of financial time series. Studies

by Mostafa et al. (2021) and Naimy et al. (2021) demonstrate that advanced GARCH models like the GJR-GARCH are effective in predicting cryptocurrency volatilities. The model specification is as follows:

$$\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2 + \gamma\varepsilon_{t-1}^2 \mathbb{1}\{\varepsilon_{t-1} \leq 0\} \tag{3}$$

where

- $\mathbb{1}\{\varepsilon_{t-1} \leq 0\}$ is an indicator function that allows the model to differentiate the impact of positive and negative shocks on future volatility.
- ω, α, β, and γ are parameters to be estimated. These can be estimated using maximum likelihood estimation.

4.3 HAR-RV

The HAR-RV model, introduced by Corsi (2009), extends the volatility modelling by incorporating the realized volatility over different time horizons. This model captures the persistent and heterogeneous nature of volatility, acknowledging that market participants operate and react on different time scales. The HAR-RV model is widely used for modelling and forecasting realized volatility due to its ability to incorporate long memory and its straightforward estimation process. As demonstrated by Bergsli et al. (2022), the HAR-RV model has been highly effective in capturing the complex dynamics of cryptocurrency volatility. The model specification for the HAR-RV model can be expressed as follows:

$$RV_{t+1} = \beta_0 + \beta_1 RV_t^{(d)} + \beta_2 RV_t^{(w)} + \beta_3 RV_t^{(m)} + \zeta_{t+1} \tag{4}$$

where

- RV_{t+1} is the realized volatility at time $t+1$,
- $RV_t^{(d)}$ is the daily realized volatility at time t,
- $RV_t^{(w)}$ is the weekly realized volatility, typically averaged over the past 5 days,
- $RV_t^{(m)}$ is the monthly realized volatility, typically averaged over the past 22 days,
- β_0, β_1, β_2, and β_3 are parameters to be estimated. These parameters can be estimated using ordinary least squares (OLS) regression.
- ζ_{t+1} is the error term at time $t+1$.

4.4 Random Forests

Random Forests, as introduced by Breiman (2001), are an ensemble learning method that operates by constructing a multitude of decision trees during training and outputting the mean prediction of the individual trees.

As shown by Bouri et al. (2021), Random Forests are highly useful for forecasting the volatility of cryptocurrencies, as they can handle complex, non-linear relationships and can incorporate a wide range of predictors which can make them superior to traditional econometric models.

The model operates as follows: Given training data $(X_1, RV_1), \ldots, (X_n, RV_n)$, consider estimating the realized volatility RV with mean function $RV(x_0) = \mathbb{E}[RV \mid X = x_0]$ for a test point x_0 using Random Forests as an ensemble method. A Random Forest consists of B trees, and for each tree T_b (where $b \in 1, \ldots, B$), we identify the leaf $L_b(x_0)$ with the predicted response $\widehat{RV}_b(x_0)$. The prediction is then based on averaging these individual tree predictions:

$$\widehat{RV}(x_0) = \frac{1}{B} \sum_{b=1}^{B} \widehat{RV}_b(x_0)$$

However, there is an alternative angle by viewing random forests as adaptive weight generators:

$$\widehat{RV}(x_0) = \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{n} RV_i \frac{\mathbb{1}\{X_i \in L_b(x_0)\}}{|L_b(x_0)|} = \sum_{i=1}^{n} RV_i \frac{1}{B} \sum_{b=1}^{B} \frac{\mathbb{1}\{X_i \in L_b(x_0)\}}{|L_b(x_0)|} = \sum_{i=1}^{n} \alpha_i(x_0) RV_i \tag{5}$$

where

$$\alpha_i(x_0) = \frac{1}{B} \sum_{b=1}^{B} \frac{\mathbb{1}\{X_i \in L_b(x_0)\}}{|L_b(x_0)|} \tag{6}$$

5 Methodology

This section addresses several topics. First, it introduces Local Linear Forests based on the proposal by Friedberg et al. (2020), which forms the foundation of this study. Next, it lays the basis for the simulation study, which motivates our choice to use LLFs for forecasting crypto volatility. Following this, the groundwork is set for the examination of LLFs. This is done by comparison of forecast accuracies, but also by quantifying utility benefits as done by Bollerslev et al. (2018). Furthermore, we analyze the superiority of LLFs by Model Confidence Sets as proposed by Hansen et al. (2011). Lastly, we analyze the performance of LLFs in different market cycles, and the difference in performance for large-cap coins compared to mid-cap coins.

5.1 Local Linear Forests

Local Linear Forests extend the Random Forest algorithm by incorporating local linear models within each tree node. Unlike Random Forests, which make predictions based solely on the average of the leaf nodes, LLFs utilize local linear regression models to make predictions in the vicinity of each data point. This approach allows LLFs to capture complex and nonlinear relationships in the data while retaining the interpretability and robustness of Random Forests.

To create weights that can be used as a kernel for local linear regression, Local Linear Forests employ a Random Forest. Local Linear Forests, in their most basic form, take the forest weights $\alpha_i(x_0)$ from Equation 5, and use them for local regression:

$$\begin{pmatrix} \widehat{RV}(x_0) \\ \hat{\theta}(x_0) \end{pmatrix} = \underset{\mu,\theta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \alpha_i(x_0)(RV_i - \widehat{RV}(x_0) - (X_i - x_0)\theta(x_0))^2 + \lambda ||\theta(x_0)||_2^2 \right\} \tag{7}$$

where

- $\widehat{RV}(x_0)$ estimates the realized volatility $RV(x_0)$,
- $\theta(x_0)$ corrects for the local trend in $X_i - x$
- $\lambda||\theta(x_0)||_2^2$, the ridge penalty, prevents overfitting.

This optimization problem has a closed-form solution:

$$\begin{pmatrix} \widehat{RV}(x_0) \\ \hat{\theta}(x_0) \end{pmatrix} = (\Delta^T A \Delta + \lambda J)^{-1} \Delta^T A \cdot RV \tag{8}$$

where

- Δ is the centered regression matrix with intercept: $\Delta_{i,1} = 1$ and $\Delta_{i,j+1} = x_{i,j} - x_{0,j}$
- A is the diagonal matrix with $A_{i,i} = \alpha_i(x_0)$
- J is the $(d+1) \times (d+1)$ diagonal matrix with $J_{1,1} = 0$ and $J_{i+1,i+1} = 1$,as the intercept is not penalized.

Therefore, the LLF weighted estimator for $\mu(x_0)$ becomes:

$$\widehat{RV}(x_0) = \sum_{i=1}^{n} \gamma_i \alpha_i(x_0) RV_i \tag{9}$$

where $\gamma_i = e_i(\Delta^T A \Delta + \lambda J)^{-1} \Delta^T$

5.2 Simulation Study

This research will conduct a simulation study, similar to one performed by Friedberg et al. (2020) to assess the performance of LLF, which will be compared with the Random Forest, Lasso Random Forest, BART and XGBoost. The comparison is based on the Root Mean Square Error (RMSE) and QLIKE (Patton, 2011).

5.2.1 Simulation 1 - GJR-GARCH

As shown by Mostafa et al. (2021) and Naimy et al. (2021), advanced GARCH models such as GJR-GARCH are well-performing models for predicting cryptocurrency volatilities. Therefore, data is simulated according to the GJR-GARCH framework to examine the performance of Local Linear Forests in this setting compared to other well-established Machine Learning models. Simulating data under the GJR-GARCH framework allows us to evaluate LLF's predictive capabilities across different volatility scenarios. The formula is as follows:

$$\sigma_t^2 = \omega + \alpha \varepsilon_{t-1}^2 + \beta \sigma_{t-1}^2 + \gamma \varepsilon_{t-1}^2 \mathbb{1}\{\varepsilon_{t-1} \leq 0\} \tag{10}$$

The simulation parameters are based on the study by Karimi et al. (2023) which estimated Bitcoin volatility using GJR-GARCH. The data given to the models are five lagged values of σ_t^2. Additionally, the dimension $d \in \{0, 5, 10\}$ varies to create some noise, while having the sample size $n \in \{1000, 5000\}$. Three scenarios are examined with error standard deviation $\sigma \in \{1, 5, 10\}$.

5.2.2 Simulation 2 - Friedman's Example

For the second simulation, an example from Friedman (1991) is employed. This model represents a popular method for examining non-parametric regression methods. First, we generate $X_1, \ldots, X_n$ independently and identically distributed from $U[0,1]^d$, and then model Y_i as follows:

$$y_i = 10\sin(\pi X_{i1} X_{i2}) + 20(X_{i3} - 0.5)^2 + 10X_{i4} + 5X_{i5} + \varepsilon,\ \varepsilon \sim N(0, \sigma^2) \tag{11}$$

This equation is particularly valuable for volatility forecasting because it tests the ability to capture nonlinear and many-sided patterns inherent in volatility data. For instance, the sinusoidal term $\sin(\pi X_{i1} X_{i2})$ evaluates whether the model can detect cyclic or periodic components, which are common in volatility trends. Additionally, the quadratic term $20(X_{i3} - 0.5)^2$ assesses LLF's capability to identify sudden shifts. Lastly, the linear signal $10X_{i4} + 5X_{i5}$ evaluates how well the model handles straightforward linear relationships, essential for capturing gradual changes or trends in volatility over time. Together, these components provide a comprehensive evaluation of LLF's robustness for volatility forecasting tasks.

The dimension $d \in \{10, 30, 50\}$ varies while having the sample size $n \in \{1000, 5000\}$. Two scenarios are examined with error standard deviation $\sigma \in \{5, 20\}$.

5.2.3 Simulation 3 - Smoothness and Local Trends

For the final simulation, we generate $X_1, \ldots, X_n$ independently and identically distributed (i.i.d.) from $U[0,1]^d$, and model responses according the following equation:

$$y_i = \log(1 + \exp(6X_{i1})) + \varepsilon,\ \varepsilon \sim N(0, \sigma^2) \tag{12}$$

This simulation examines the effectiveness of handling smoothness in the regression surface and sees whether the models can capture strong local trends accurately. By focusing on these specific aspects, we examine whether LLF can effectively model the nuanced and rapidly changing patterns of volatility observed in financial data. The models are tested on a grid with dimension $d \in \{5, 20\}$, sample size $n \in \{1000, 5000\}$, and error standard deviation $\sigma \in \{0.1, 1, 2\}$.

5.3 Forecasting Accuracy Comparison

This study adopts a methodical approach to assess the forecasting efficacy of Local Linear Forests (LLFs) compared to established baseline models, namely GARCH(1,1), GJR-GARCH, HAR-RV and Random Forest. Our evaluation encompasses both in-sample fit and out-of-sample forecast accuracy, employing standard evaluation metrics such as RMSE, Mean Absolute Error (MAE) and QLIKE (Patton, 2011). For the out-of-sample forecasts, the model is trained on seventy per cent of the data. Then forecasts are made for the remaining thirty per cent. By adhering to established evaluation standards, we strive to provide empirical evidence regarding the effectiveness of LLFs in capturing the complexities of financial markets.

5.4 Quantifying Utility Benefits

This study also implements a method to quantify the utility benefits of various volatility forecasting models for risk-targeting investors. Our approach builds on the framework introduced by Bollerslev et al. (2018), which provides a robust method for evaluating the utility of different volatility models within the context of mean-variance preferences.

The principle behind this methodology is to invest in an asset with time-varying volatility while maintaining a constant Sharpe ratio (SR). Specifically, an investor with wealth W_t and risk-aversion parameter γ allocates a portion x_t of their wealth to a risky asset with return r_{t+1}, and the remaining $(1 - x_t)$ to a risk-free asset. The utility derived from this allocation is expressed as:

$$U_t(x_t) = W_t \left(x_t \cdot \mathbb{E}_t(r_{t+1}) - \frac{\gamma}{2} \cdot x_t^2 \cdot \mathbb{E}_t(RV_{t+1}) \right) \tag{13}$$

which can be rewritten if a constant conditional Sharpe ratio is assumed: $SR = \frac{\mathbb{E}_t(r_{t+1})}{\sqrt{\mathbb{E}_t(RV_{t+1})}}$, leading to:

$$U_t(x_t) = W_t \left(x_t \cdot SR \cdot \sqrt{\mathbb{E}_t(RV_{t+1})} - \frac{\gamma}{2} \cdot x_t^2 \cdot \mathbb{E}_t(RV_{t+1}) \right) \tag{14}$$

The optimal weight allocation is then given by:

$$x_t^* = \frac{SR/\gamma}{\sqrt{\mathbb{E}_t(RV_{t+1})}} \tag{15}$$

We consider the expected utility per unit of wealth for a mean-variance investor with risk aversion parameter $\gamma = 2$, an annualized Sharpe Ratio of 0.4, and a target annualized volatility of 20%. Now let $\mathbb{E}_t^\theta[\cdot]$ equal the expectations from model θ and let $\mathbb{E}_t[\cdot]$ equal the expectations from the real (unknown) risk model. Then the expected utility per unit of wealth is:

$$U_0 W_t = \frac{U(x_t^\theta)}{W_t^\theta} = 8\% \frac{\sqrt{\mathbb{E}_t(RV_{t+1})}}{\sqrt{\mathbb{E}_t^\theta(RV_{t+1})}} - 4\% \frac{\mathbb{E}_t(RV_{t+1})}{\mathbb{E}_t^\theta(RV_{t+1})} \tag{16}$$

We evaluate this expected utility by taking the average of the realized expressions over the same forecasts:

$$U_0 W^\theta = \frac{1}{T} \sum_{t=1}^{T} 8\% \frac{\sqrt{RV_{t+1}}}{\sqrt{\mathbb{E}_t^\theta(RV_{t+1})}} - 4\% \frac{RV_{t+1}}{\mathbb{E}_t^\theta(RV_{t+1})} \tag{17}$$

A perfect forecast would result in a realized utility of 4%. By implementing this comprehensive methodology, we assess the utility benefits of different volatility forecasting models, thereby aiding risk-targeting investors in making informed investment decisions.

5.5 Model Confidence Set

To test whether the improvements in forecasting performance are significant, we utilize the Model Confidence Set (MCS) approach (Hansen et al., 2011). We test this across all loss functions introduced in this paper, that is, RMSE, MAE, QLIKE and the (negative) expected utility.

Let M be the set of all models. For each coin i and loss function L, we define

$$d_{i,t+1}^{j,k} = L\left(RV_{t+1}, \widehat{RV}_{t+1}^{j}\right) - L\left(RV_{t+1}, \widehat{RV}_{t+1}^{k}\right) \tag{18}$$

as the difference in loss between model j and model k. We calculate the average difference in loss per coin, $\bar{d}_i^{j,k}$. From here, we calculate the test statistic:

$$t_i^{j,k} = \frac{\bar{d}_i^{j,k}}{\widehat{Var}(d_i^{j,k})} \forall j,k \in M \tag{19}$$

The MCS test statistic is given by $T_{i,M} = \max_{j,k \in M} \left| t_i^{j,k} \right|$, which has a null hypothesis that all models have the same loss, meaning all models perform equally well. If the null hypothesis is rejected, the worst-performing model is taken out of the set. This is done iteratively until the null is no longer rejected. The final set of models for coin i is given by MCS_i and is the set of best-performing models with confidence $1 - \alpha$. Supported by literature (Kleen and Tetereva, 2022; Laurent et al., 2013; Liu et al., 2015), we choose $\alpha = 0.1$.

We approximate the distribution of $T_{i,M}$ by block-bootstrapping, as proposed by Hansen et al. (2011) with a block length of 7. To measure the overall performance of the models, over all coins, we calculate the share of coins for which model j is in the Model Confidence Set:

$$MCSR^j = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\{j \in MCS_i\} \tag{20}$$

5.6 Market Cycles

This study investigates the variability in forecast performance across different market phases, namely bull, bear, and consolidating periods.

First, market phases are defined based on observed price movements using a rolling window of thirty days. This segmentation allows us to categorize periods into bull phases, characterized by rising prices (more than five per cent increase), bear phases marked by declining prices (more than five per cent decrease), and consolidating phases represented by stable price movements.

To assess the forecastability of volatility across market phases, we would like to conduct ANOVA (Fisher, 1970) and Tukey's Honest Significant Difference (HSD) tests (Tukey, 1949) on the squared forecast errors. However, as these assume a normal distribution, we will use non-parametric alternatives instead, namely the Kruskal-Wallis test (Kruskal and Wallis, 1952) and Dunn's Test (Dunn, 1964). The Kruskal-Wallis test is a non-parametric method for testing whether samples originate from the same distribution. It can be used to determine if there are statistically significant differences in forecast errors among the market phases. Dunn's test is a post-hoc analysis used after a Kruskal-Wallis test to determine which specific phases are different. It allows for multiple pairwise comparisons while controlling for Type I errors.

If certain periods exhibit significantly lower forecast errors, it implies that volatility is more predictable during these phases, enhancing the reliability of forecasting models. Understanding these dynamics can inform the selection and improvement of forecasting tools, thereby improving financial risk management and investment strategies.

5.7 Large-cap vs. Mid-cap Coins

The categorization of the cryptocurrency dataset into distinct segments based on market capitalization enables us to explore potential differences in the volatility forecasts between large-cap and mid-cap cryptocurrencies. Large-cap cryptocurrencies, such as Bitcoin and Ethereum, typically dominate the market in terms of market capitalization and trading volume. On the other hand, mid-cap cryptocurrencies represent a diverse collection of smaller, less-established digital assets with market capitalizations ranging between 5 and 10 billion USD.

Understanding whether volatility forecasts differ between large-cap and mid-cap coins can provide insights into the dynamics of market behaviour and risk across different segments of the cryptocurrency landscape. Additionally, investors and traders may have distinct risk preferences or strategies when dealing with large-cap versus mid-cap cryptocurrencies. Therefore, identifying any significant differences in volatility forecasts can inform risk management practices and investment decisions tailored to specific market segments.

The intuition behind investigating differences in volatility forecasts lies in the inherent characteristics and market dynamics of large-cap and mid-cap cryptocurrencies. Large-cap cryptocurrencies, with their higher market capitalization and liquidity, may exhibit more stable and predictable price movements compared to mid-cap coins. On the other hand, mid-cap cryptocurrencies, being relatively smaller and less liquid, may experience higher volatility and greater susceptibility to market fluctuations and speculative activity. Therefore, it is plausible to hypothesize that volatility forecasts for mid-cap cryptocurrencies could differ significantly from those of their large-cap counterparts.

To test this hypothesis, forecast errors for both groups are obtained. To ensure comparability across datasets, the errors are normalized with the squared logarithmic error (SLE), defined as follows:

$$SLE_t = \left(log\left(\frac{f_t}{RV_t}\right)\right)^2 \tag{21}$$

We utilize this measure instead of the simple squared forecast error, as this SLE is scale-invariant. This allows us to use the Root Mean Square Log Error (RMSLE), which is given as follows:

$$RMSLE = \sqrt{\frac{1}{T}\sum_{i=1}^{T} SLE_t} \tag{22}$$

where f_t is the forecasted value for realized volatility RV_t at time t. The RMSLE is scale-invariant, which means it can be used as a comparison measure across datasets, or in our case, across groups of cryptocurrencies.

Using the RMSLE, the standard deviation of the SLEs and the skewness of the SLEs, we will examine whether a difference can be found in the accuracy of forecasts for large-cap cryptocurrencies compared to mid-cap coins.

6 Results

This section presents the outcomes of our empirical analyses, focusing on the performance of Local Linear Forests in forecasting cryptocurrency market volatility. We begin with a detailed simulation study to assess the robustness of LLFs. Following this, we compare the LLFs' performance against established models such as GARCH, GJR-GARCH, HAR-RV, and Random Forests. We compare them across digital currencies, based on forecasting errors and realized utility. Furthermore, we compare them across market cycles and segments. Finally, we look at the feature importance for Local Linear Forests.

6.1 Simulation Study

In this section, we analyze the results from our simulations. Detailed findings from Simulation 2 and Simulation 3 are available in the Appendix, where the hyperparameters of all models have been tuned in advance. Details on the tuning can also be found in the Appendix.

6.1.1 Simulation 1 - GJR-GARCH

The results of the first simulation can be found in Table 1 and 2.

Table 1: Root Mean Squared Error for Simulation 1

d	n	σ	**LLF**	**RF**	**Lasso-RF**	**BART**	**XGBoost**
0	1000	1	0.130	**0.045**	0.057	0.053	0.056
0	5000	1	0.099	0.051	0.047	0.054	**0.033**
5	1000	1	0.165	0.083	0.088	0.085	**0.082**
5	5000	1	0.139	0.042	0.043	0.043	0.027
10	1000	1	0.162	0.076	0.086	0.104	**0.047**
10	5000	1	**0.054**	0.057	0.060	**0.054**	0.063
0	1000	5	0.320	**0.204**	0.251	0.230	0.230
0	5000	5	**0.430**	0.508	0.483	0.476	0.512
5	1000	5	0.352	**0.211**	0.293	0.228	0.250
5	5000	5	0.378	0.443	**0.235**	0.384	0.378
10	1000	5	0.321	**0.237**	0.308	0.284	0.239
10	5000	5	**0.264**	0.283	0.271	0.345	0.297
0	1000	10	0.839	0.737	0.793	**0.646**	0.718
0	5000	10	**0.317**	0.429	0.421	0.598	0.815
5	1000	10	**0.632**	0.956	0.832	0.712	0.723
5	5000	10	**0.526**	0.527	0.638	0.495	0.544
10	1000	10	**0.402**	0.602	0.756	0.717	0.464
10	5000	10	**0.327**	0.339	0.394	0.385	0.449

Notes: The data ranges over dimensions $d \in \{0, 5, 10\}$, sample size $n \in \{1000, 5000\}$ and error standard deviation $\sigma \in \{1, 5, 10\}$. The lowest prediction error is highlighted in bold for each case.

Table 2: QLIKE for Simulation 1

d	n	σ	**LLF**	**RF**	**Lasso-RF**	**BART**	**XGBoost**
0	1000	1	0.124	**0.041**	0.053	0.051	0.053
0	5000	1	0.108	0.061	0.056	0.058	**0.039**
5	1000	1	0.165	0.081	0.087	0.080	**0.076**
5	5000	1	0.140	0.050	0.052	0.046	**0.027**
10	1000	1	0.148	0.064	0.077	0.092	**0.045**
10	5000	1	**0.054**	0.059	0.061	0.056	0.068
0	1000	5	0.069	**0.046**	0.054	0.051	0.048
0	5000	5	**0.084**	0.103	0.098	0.091	0.104
5	1000	5	0.070	**0.040**	0.054	0.044	0.049
5	5000	5	0.083	0.088	**0.057**	0.073	0.069
10	1000	5	0.069	**0.049**	0.065	0.064	0.052
10	5000	5	**0.055**	0.056	0.059	0.068	0.071
0	1000	10	0.095	0.084	0.088	0.074	**0.072**
0	5000	10	**0.031**	0.045	0.049	0.061	0.089
5	1000	10	**0.066**	0.082	0.074	0.075	0.070
5	5000	10	0.056	0.057	0.063	**0.053**	0.058
10	1000	10	**0.037**	0.057	0.065	0.058	0.042
10	5000	10	**0.035**	0.037	0.038	0.036	0.040

Notes: The data ranges over dimensions $d \in \{0, 5, 10\}$, sample size $n \in \{1000, 5000\}$ and error standard deviation $\sigma \in \{1, 5, 10\}$. The lowest prediction error is highlighted in bold for each case.

In lower-noise environments, LLF consistently underperforms, both in terms of RMSE and QLIKE. It often emerges as the worst model, especially in lower dimensions. However, as noise and dimensions increase, LLF's superiority becomes pronounced. It frequently outperforms other models across different dimensions and sample sizes. This robustness under noise shows LLF's capacity to handle cryptocurrency volatility data with inherent uncertainties, which are known for their noisy environment.

6.2 Forecasting Accuracy

In this section, we analyze the results derived from our forecasts. Details on the tuning of the hyperparameters of Local Linear Forest and Random Forest can be found in the Appendix. An analysis of the in-sample predictions can be found in the Appendix. The results of the out-of-sample predictions are found in Tables 3, 4 and 5.

Table 3: Out-Of-Sample Root-Mean-Squared Errors

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Bitcoin (BTC)	**0.541**	0.705	1.324	1.293	1.250
Ethereum (ETH)	**0.256**	0.724	1.450	1.448	1.365
Tether (USDT)	**0.244**	0.248	0.407	0.408	0.427
Binance Coin (BNB)	**0.676**	0.750	1.473	1.499	1.462
Bitcoin Cash (BCH)	**0.745**	1.227	2.859	3.103	2.190
Litecoin (LTC)	**0.608**	0.899	1.862	1.868	1.826
Internet Computer (ICP)	**0.705**	1.856	2.591	2.642	2.536
Polygon (MATIC)	**0.807**	0.908	1.761	1.760	1.738

Notes: The lowest RMSEs are highlighted in bold for each currency.

Table 4: Out-Of-Sample Mean-Absolute Errors

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Bitcoin (BTC)	**0.377**	0.432	1.066	1.020	0.882
Ethereum (ETH)	**0.199**	0.448	1.077	1.073	0.926
Tether (USDT)	0.137	**0.101**	0.221	0.220	0.235
Binance Coin (BNB)	**0.446**	0.452	1.031	1.059	0.947
Bitcoin Cash (BCH)	**0.567**	0.835	1.711	1.760	1.550
Litecoin (LTC)	**0.458**	0.555	1.415	1.424	1.202
Internet Computer (ICP)	**0.519**	1.085	1.591	1.637	1.579
Polygon (MATIC)	**0.612**	0.638	1.319	1.320	1.225

Notes: The lowest MAEs are highlighted in bold for each currency.

Table 5: QLIKE Out-of-Sample

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Bitcoin (BTC)	0.061	**0.031**	0.168	0.163	0.156
Ethereum (ETH)	**0.009**	0.026	0.138	0.142	0.125
Tether (USDT)	0.144	**0.048**	0.289	0.294	0.299
Binance Coin (BNB)	0.030	**0.028**	0.129	0.136	0.127
Bitcoin Cash (BCH)	**0.020**	0.037	0.130	0.132	0.144
Litecoin (LTC)	**0.020**	0.025	0.142	0.142	0.128
Internet Computer (ICP)	**0.014**	0.038	0.096	0.097	0.084
Polygon (MATIC)	0.027	**0.025**	0.099	0.099	0.092

Notes: The lowest QLIKEs are highlighted in bold for each currency.

LLF consistently demonstrates superior performance compared to the baseline models across all cryptocurrencies in terms of RMSE and MAE. The only model that is able to offer competition is the Random Forest, but in most cases, LLF clearly outperforms. The GARCH-type and HAR-RV models have comparable performances, with HAR-RV being slightly better. However, none of them come close to the LLF and RF. In terms of QLIKE, Random Forest pulls closer to LLF's performance and even outperforms it about half the time. As QLIKE is less sensitive to extreme errors, this implies that Random Forest in general has more extreme errors. But if these are not excessively punished - as the squared error does - the performance of LLF and RF is comparable. The only cryptocurrency we see that RF outperforms LLF is Tether. This could be due to Tether's nature as a stablecoin, which results in low volatility and predictable price movements, which are better captured by Random Forest's robust, ensemble approach. In contrast, the added complexity and local sensitivity of Local Linear Forests, which excel in more volatile environments, do not provide significant advantages for the stable and smooth price patterns of Tether.

6.3 Quantifying Utility Benefits

Table 6 shows the results of quantifying the realized utility benefits.

Table 6: Realized Utility Benefits

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**	**Future RV**
Bitcoin (BTC)	3.85%	**3.94%**	3.71%	3.71%	3.69%	4.00%
Ethereum (ETH)	**3.98%**	3.95%	3.74%	3.73%	3.74%	4.00%
Tether (USDT)	3.77%	**3.91%**	3.46%	3.44%	3.44%	4.00%
Binance Coin (BNB)	3.94%	**3.95%**	3.75%	3.74%	3.74%	4.00%
Bitcoin Cash (BCH)	**3.96%**	3.93%	3.75%	3.74%	3.69%	4.00%
Litecoin (LTC)	**3.96%**	3.95%	3.74%	3.74%	3.73%	4.00%
Internet Computer (ICP)	**3.97%**	3.92%	3.79%	3.83%	3.09%	4.00%
Polygon (MATIC)	**3.95%**	**3.95%**	3.81%	3.81%	3.81%	4.00%

Notes: The highest realized utility benefits are highlighted in bold for each currency. The column labelled "Future RV" reports the utility from exactly knowing the future volatility.

Across the evaluated models, LLF demonstrates a very strong performance, often coming close to the "perfect" forecast, having a realized utility benefit of 3.95% or greater for five out of the eight cryptocurrencies. There is a clear distinction between LLF and the GARCH-type and HAR-RV models. Equivalent to the forecasting results previous section, only Random Forest is able to offer LLF competition. There is one currency where RF clearly outperforms LLF (Tether), but this is likely again due to Tether's nature as a stablecoin. These superior utility benefit results are another measure that shows that LLF could be a new preferred tool for volatility forecasting.

6.4 Model Confidence Set

The models' results with respect to the MCS Rate statistic are listed in Table 7.

Table 7: Model Confidence Set Rates

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
RMSE	**1.000**	0.375	0.000	0.000	0.000
MAE	**0.875**	0.375	0.000	0.000	0.000
QLIKE	**0.875**	0.500	0.000	0.000	0.000
(Negative) Utility	**0.875**	0.500	0.000	0.000	0.000

Notes: MCSRs at $\alpha = 0.1$ are noted for each loss function utilized in our analysis. Utility is made negative as MCS looks at minimizing loss functions. The highest MCSRs are highlighted in bold for each loss function.

We observe a very clear distinction. While Random Forest is able to offer some competition to LLF in some of the measures, such as QLIKE and (negative) utility, overall LLF outperforms Random Forest. As we could already observe in the earlier sections, the GARCH-type models and HAR-RV are not able to compete with the Local Linear Forest when it comes to forecasting volatility. As we can observe, for MAE, QLIKE and (negative) utility, there is only one coin where LLF is not included in the superior models. This is Tether, as seen in previous sections. This shows that while LLF is generally the most effective model, the unique characteristics of stablecoins like Tether can alter the performance dynamics of forecasting models.

6.5 Market Cycles

The results of the analysis across market cycles for Bitcoin can be found in Table 8. The results for the analyses of other cryptocurrencies can be found in the Appendix. Tether is excluded from this analysis, due to it being a stablecoin (explanation can be found in the Appendix).

Across all currencies examined, the findings consistently show that the LLF model delivers the most accurate forecasts across every phase of the market cycle. This is particularly made clear by the Model Confidence Set, which can be found in Table 20. This demonstrates LLF's ability to maintain consistent performance without losing in specific market phases.

Another noteworthy observation pertains to the considerable disparity in forecasting errors across different market phases. The mean squared forecast errors during bearish and consolidating periods appear notably lower compared to bullish periods. Moreover, there is a substantial difference in standard deviation, with volatility in consolidating phases exhibiting easier predictability, likely owing to the more stable nature of returns during these intervals. Conversely, bull phases prove to be the most challenging to predict. The significance of these periodic differences is assessed using the Kruskal-Wallis test, revealing that LLF demonstrates significant results for only three out of the seven currencies tested. In contrast, Random Forest and HAR-RV exhibit significant results across all currencies, while GARCH-type models show significant results for three and five currencies, respectively.

To assess specific differences between phases, Dunn's test is employed. As anticipated, significant differences primarily emerge between consolidating and bull phases, with occasional

significant differences also observed between bear and bull phases.

Table 8: Statistical Summary of Squared Forecast Errors Across Market Cycles for Bitcoin.

		LLF	RF	GARCH	GJR-GARCH	HAR-RV
Mean *(St. Dev.)*	Bear	**0.268**	0.293	2.235	2.317	1.880
		(0.44)	*(0.79)*	*(2.32)*	*(2.24)*	*(3.55)*
	Consolidating	**0.194**	0.264	1.532	1.424	1.009
		(0.40)	*(0.67)*	*(1.78)*	*(1.70)*	*(2.22)*
	Bull	**0.447**	0.899	1.964	1.883	2.315
		(1.17)	*(4.98)*	*(5.61)*	*(6.51)*	*(7.57)*
Kruskal-Wallis	H-statistic	8.16	6.69	4.28	9.22	8.76
	P-value	0.017*	0.035*	0.117	0.010*	0.013*
Dunn	Bear-Bull	1.000	0.831	0.122	0.008*	1.000
	Bear-Consolidating	0.409	1.000	0.382	0.087	0.730
	Bull-Consolidating	0.019*	0.030*	0.955	0.379	0.011*

Notes: For Dunn's test, the p-value is noted. Significance at the $\alpha = 0.05$ level is indicated by an asterisk (*). The lowest mean squared forecast errors are highlighted in bold.

The significant differences identified by the Kruskal-Wallis and Dunn's tests indicate that market phases critically influence the predictability of volatility. Specifically, volatility demonstrates higher predictability during consolidating phases compared to bull phases, as evidenced by lower forecast errors in those periods. Notably, LLF and GARCH models exhibit minimal sensitivity to changes in market phases, underscoring their robustness. Combined with LLFs consistently low squared errors, it attests to its reliability.

6.6 Large-cap vs. Mid-cap

The comparison of large-cap and mid-cap cryptocurrency forecasts can be found in Table 9.

Table 9: Statistical Summary of Relative Forecast Errors for Large-cap and Mid-cap Cryptocurrencies

		LLF	RF	GARCH (1,1)	GJR-GARCH	HAR-RV
Large-cap	*RMSLE*	0.382	**0.267**	0.663	0.663	0.620
	St. Dev.	0.498	0.140	0.788	0.788	0.767
	Skewness	6.224	6.000	3.619	3.605	4.626
Mid-cap	*RMSLE*	**0.207**	0.254	0.503	0.506	0.460
	St. Dev.	0.072	0.108	0.377	0.381	0.345
	Skewness	4.067	4.049	3.474	3.705	4.079

Notes: The lowest RMSLEs are highlighted in bold.

The RMSLE values consistently indicate lower values for mid-cap cryptocurrencies compared to large-cap cryptocurrencies across all models. This suggests that volatility forecasts for mid-

cap cryptocurrencies are more accurate, implying greater predictability. This phenomenon may be attributed to the potential lower market efficiency of mid-cap cryptocurrencies relative to large-cap currencies, allowing for more predictable price movements. In less efficient markets, there exists a lag between the arrival of new information and its full integration into prices, which provides opportunities for forecasting models to identify and exploit patterns or trends.

The standard deviation of forecast errors is similarly lower for mid-cap cryptocurrencies, indicating that forecasts for mid-caps are more consistent and consequently more reliable.

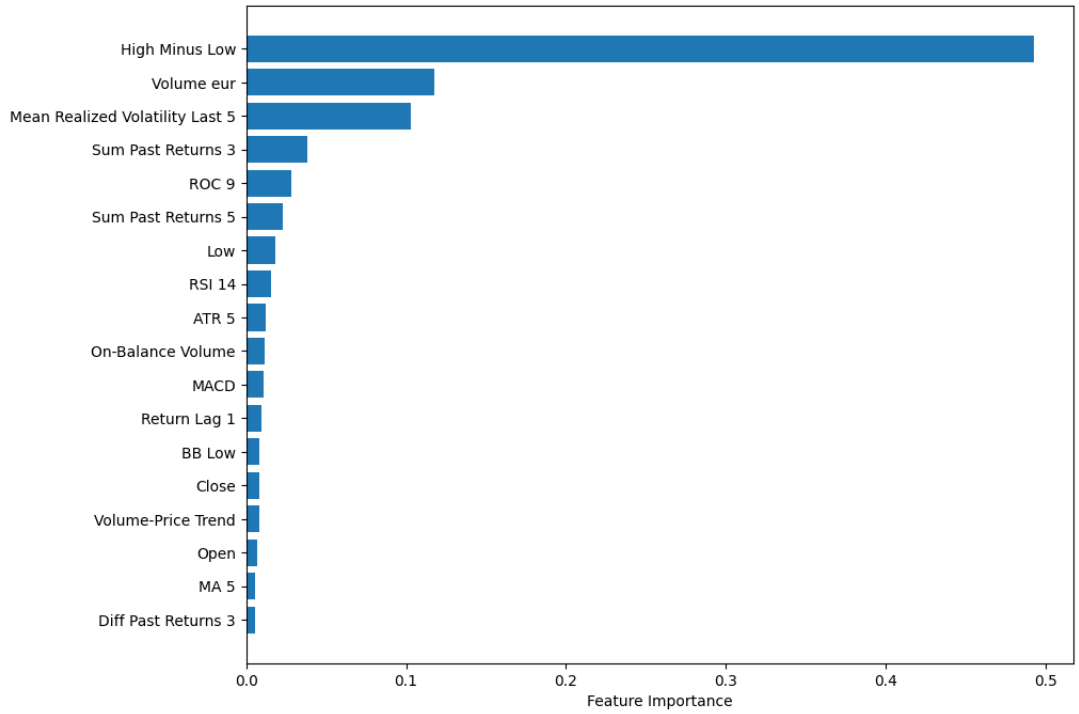
Conversely, large-cap cryptocurrencies exhibit higher skewness in the distribution of relative forecast errors for non-GARCH models. This indicates these models possess a thicker right tail, implying a more frequent occurrence of extreme forecast errors compared to mid-cap coins.

The combination of the RMSLE and the standard deviation of the normalized forecast errors indicate that forecasts for mid-cap cryptocurrencies seem to be more reliable and consistently performing than the forecasts for large-cap cryptocurrencies.

6.7 Feature Importance

The importance of each feature for LLF in the case of forecasting Bitcoin's volatility can be found in Figure 1. For the other cryptocurrencies, the feature importance figures can be found in the Appendix.

Figure 1: Bitcoin Feature Importance for Local Linear Forests



Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

A consistent pattern emerges across most coins regarding the key features that hold significant importance. Notably, 'High Minus Low', 'Volume in Euros', the first lags of 'Return' and 'Realized Volatility', and the 'Mean of the last five Realized Volatilities' stand out as crucial

indicators. These metrics, despite their relative simplicity compared to more complex technical features, play key roles in the forecasts made by LLF.

The observation that relatively simple measures hold more importance than complex technical indicators in forecasting cryptocurrency volatility is intriguing for several reasons. Firstly, it challenges the assumption that sophisticated models and intricate algorithms are always necessary to capture the dynamics of volatile markets like cryptocurrencies. Instead, it suggests that basic, easily interpretable metrics can provide substantial predictive power. Furthermore, the preference for simple metrics highlights a broader trend in quantitative finance and data science, where there is a growing recognition that simplicity can often lead to more robust and interpretable models (Hansen, 2020).

7 Conclusion

In conclusion, this study has rigorously assessed the efficiency of Local Linear Forests as a novel volatility forecasting model through means of a simulation study and several forecasting analyses. We contrast LLFs with established benchmarks such as GARCH, GJR-GARCH, HAR-RV, and Random Forests. Our comprehensive analysis reveals that LLFs consistently outperform these baseline models in volatility forecasting. Moreover, beyond their superior forecasting accuracy, LLFs demonstrate notable utility benefits, yielding the highest realized utility for the majority of cryptocurrencies. This suggests that LLFs not only enhance predictive accuracy but also contribute positively to risk-adjusted returns and portfolio optimization strategies in volatile market conditions. Furthermore, we observe significant variations in forecasting errors across different market phases, with bull markets presenting greater challenges for volatility prediction compared to consolidating periods, although LLFs are less affected by this compared to the baseline models and are superior in each phase. In our analysis of market segments, mid-cap cryptocurrencies exhibit relatively lower forecast errors than their large-cap counterparts, highlighting the nuanced dynamics within cryptocurrency markets.

Additionally, our simulation study showcased LLF's versatility and adaptability. By integrating LLF into risk management frameworks, investors can better navigate market uncertainties and capitalize on emerging opportunities with greater confidence.

Future research could explore several critical avenues to enhance predictive accuracy and applicability in dynamic market conditions. Firstly, investigating how Local Linear Forests perform during extreme market events, such as regulatory changes or major economic announcements, is crucial for understanding model robustness and reliability. These events often trigger significant price movements, challenging traditional forecasting methods. Secondly, integrating alternative data sources such as sentiment analysis from social media or blockchain metrics, could enrich the predictive capabilities of LLFs by capturing non-traditional market signals that influence cryptocurrency price dynamics. Finally, future research could explore dynamic portfolio allocation strategies using volatility forecasts to optimize risk-adjusted returns over varying investment horizons. This approach involves continuously adjusting portfolio weights based on volatility forecasts, thereby adapting to changing market conditions and investor risk preferences more effectively. These directions not only aim to improve forecasting accuracy but also pave the way for more resilient risk management frameworks in cryptocurrency investments.

References

Agbo, C. C., Mahmoud, Q. H., and Eklund, J. M. (2019). Blockchain technology in healthcare: a systematic review. In *Healthcare*, volume 7, page 56. MDPI.

Akyildirim, E., Goncu, A., and Sensoy, A. (2021). Prediction of cryptocurrency returns using machine learning. *Annals of Operations Research*, 297:3–36.

Bâra, A. and Oprea, S.-V. (2024). An ensemble learning method for bitcoin price prediction based on volatility indicators and trend. *Engineering Applications of Artificial Intelligence*, 133:107991.

Bergsli, L. Ø., Lind, A. F., Molnár, P., and Polasik, M. (2022). Forecasting volatility of bitcoin. *Research in International Business and Finance*, 59:101540.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327.

Bollerslev, T., Hood, B., Huss, J., and Pedersen, L. H. (2018). Risk everywhere: Modeling and managing volatility. *The Review of Financial Studies*, 31(7):2729–2773.

Bouri, E., Gkillas, K., Gupta, R., and Pierdzioch, C. (2021). Forecasting realized volatility of bitcoin: The role of the trade war. *Computational Economics*, 57:29–53.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Catania, L. and Grassi, S. (2017). Modelling crypto-currencies financial time-series. *Available at SSRN 3028486*.

Cootner, P. H. (1964). The random character of stock market prices.

Corsi, F. (2009). A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2):174–196.

Dunn, O. J. (1964). Multiple comparisons using rank sums. *Technometrics*, 6(3):241–252.

D'Amato, V., Levantesi, S., and Piscopo, G. (2022). Deep learning in predicting cryptocurrency volatility. *Physica A: Statistical Mechanics and its Applications*, 596:127158.

Engle, R. (2001). Garch 101: The use of arch/garch models in applied econometrics. *Journal of economic perspectives*, 15(4):157–168.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007.

Fama, E. F. (1965). Random walks in stock market prices. *Financial Analysts Journal*, 21(5):55–59.

Fink, C. and Johann, T. (2014). Bitcoin markets. *Available at SSRN 2408396*.

Fisher, R. A. (1970). Statistical methods for research workers. In *Breakthroughs in statistics: Methodology and distribution*, pages 66–70. Springer.

Friedberg, R., Tibshirani, J., Athey, S., and Wager, S. (2020). Local linear forests. *Journal of Computational and Graphical Statistics*, 30(2):503–517.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, 19(1):1–67.

Ftiti, Z., Louhichi, W., and Ben Ameur, H. (2023). Cryptocurrency volatility forecasting: What can we learn from the first wave of the covid-19 outbreak? *Annals of Operations Research*, 330(1):665–690.

Glosten, L. R., Jagannathan, R., and Runkle, D. E. (1993). On the relation between the expected value and the volatility of the nominal excess return on stocks. *The journal of finance*, 48(5):1779–1801.

Gradojevic, N., Kukolj, D., Adcock, R., and Djakovic, V. (2023). Forecasting bitcoin with technical analysis: A not-so-random forest? *International Journal of Forecasting*, 39(1):1–17.

Gurtu, A. and Johny, J. (2019). Potential of blockchain technology in supply chain management: a literature review. *International Journal of Physical Distribution & Logistics Management*, 49(9):881–900.

Hansen, K. B. (2020). The virtue of simplicity: On machine learning models in algorithmic trading. *Big Data & Society*, 7(1):2053951720926558.

Hansen, P. R., Lunde, A., and Nason, J. M. (2011). The model confidence set. *Econometrica*, 79(2):453–497.

Karimi, P., Ghazani, M. M., and Ebrahimi, S. B. (2023). Analyzing spillover effects of selected cryptocurrencies on gold and brent crude oil under covid-19 pandemic: Evidence from gjr-garch and evt copula methods. *Resources Policy*, 85:103887.

Kim, J.-M., Jun, C., and Lee, J. (2021). Forecasting the volatility of the cryptocurrency market by garch and stochastic volatility. *Mathematics*, 9(14):1614.

Kleen, O. and Tetereva, A. (2022). A forest full of risk forecasts for managing volatility. *Available at SSRN 4161957*.

Kruskal, W. H. and Wallis, W. A. (1952). Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621.

Kurihara, Y. and Fukushima, A. (2017). The market efficiency of bitcoin: a weekly anomaly perspective. *Journal of Applied Finance and Banking*, 7(3):57.

Laurent, S., Rombouts, J. V., and Violante, F. (2013). On loss functions and ranking forecasting performances of multivariate volatility models. *Journal of Econometrics*, 173(1):1–10.

Liu, L. Y., Patton, A. J., and Sheppard, K. (2015). Does anything beat 5-minute rv? a comparison of realized measures across multiple asset classes. *Journal of Econometrics*, 187(1):293–311.

Liu, Y., Li, Z., Nekhili, R., and Sultan, J. (2023). Forecasting cryptocurrency returns with machine learning. *Research in International Business and Finance*, 64:101905.

Lumineau, F., Wang, W., and Schilke, O. (2021). Blockchain governance—a new way of organizing collaborations? *Organization Science*, 32(2):500–521.

Ma, F., Liang, C., Ma, Y., and Wahab, M. I. M. (2020). Cryptocurrency volatility forecasting: A markov regime-switching midas approach. *Journal of Forecasting*, 39(8):1277–1290.

Mostafa, F., Saha, P., Islam, M. R., and Nguyen, N. (2021). Gjr-garch volatility modeling under nig and ann for predicting top cryptocurrencies. *Journal of Risk and Financial Management*, 14(9):421.

Murray, K., Rossi, A., Carraro, D., and Visentin, A. (2023). On forecasting cryptocurrency prices: A comparison of machine learning, deep learning, and ensembles. *Forecasting*, 5(1):196–209.

Naimy, V., Haddad, O., Fernández-Avilés, G., and El Khoury, R. (2021). The predictive capacity

of garch-type models in measuring the volatility of crypto and world currencies. *PloS one*, 16(1):e0245904.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

Nowiński, W. and Kozma, M. (2017). How can blockchain technology disrupt the existing business models? *Entrepreneurial Business and Economics Review*, 5(3):173–188.

Patton, A. J. (2011). Volatility forecast comparison using imperfect volatility proxies. *Journal of Econometrics*, 160(1):246–256.

Said, S. E. and Dickey, D. A. (1984). Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, 71(3):599–607.

Tukey, J. W. (1949). Comparing individual means in the analysis of variance. *Biometrics*, pages 99–114.

Wang, Y., Andreeva, G., and Martin-Barragan, B. (2023). Machine learning approaches to forecasting cryptocurrency volatility: Considering internal and external determinants. *International Review of Financial Analysis*, 90:102914.

Appendix

A. Summary Statistics

Table 10: Summary Statistics of Returns (in percentages)

	Mean	Median	High	Low	St. Dev.	Skewness	Kurtosis	Observations
Bitcoin (BTC)	0.073	0.055	13.144	-15.828	2.891	-0.331	3.569	1056
Ethereum (ETH)	0.063	0.060	16.249	-18.353	3.608	-0.301	3.624	1056
Tether (USDT)	-0.008	0.000	2.062	-1.739	0.464	0.144	1.838	1056
Binance Coin (BNB)	0.068	0.094	15.795	-19.975	3.303	-0.511	4.731	1056
Bitcoin Cash (BCH)	0.002	-0.050	46.024	-18.295	4.452	1.417	15.172	1056
Litecoin (LTC)	-0.036	0.084	24.176	-21.283	4.086	-0.229	4.153	1056
Internet Computer (ICP)	-0.104	-0.011	33.734	-35.497	5.455	0.325	5.438	1056
Polygon (MATIC)	-0.029	-0.110	29.900	-28.212	5.226	0.252	4.352	1056

B. Feature Set

Table 11: Set of Features

Feature Name	**Information**
Realized Variance	Number of lags = 5
Open Price	Number of lags = 1
Close Price	Number of lags = 1
High	Number of lags = 1
Low	Number of lags = 1
High - Low	Number of lags = 1
Volume	Number of lags = 1
Returns	Number of lags = 5
Moving Average (MA)	Window = 5
Correlation between MA and Close	Number of lags = 1
Sum of Past Returns	Window = 3 & 5
Difference Past Returns: $(\sum_{k=1}^{5} r_{t-k} - \sum_{k=1}^{3} r_{t-k})$	Window = 3 & 5
Relative Strength Index (RSI)	Window = 6 & 14
$\mathbb{1}\{RSI_6 \leq 20\%\}$ & $\mathbb{1}\{RSI_{14} \leq 20\%\}$	Buy signals from RSI
$\mathbb{1}\{RSI_6 \geq 80\%\}$ & $\mathbb{1}\{RSI_{14} \leq 80\%\}$	Sell signals from RSI
MA convergence divergence (MACD)	Fast Period = 5, Slow Period = 10, Signal Period =5
Rate of Change Return	Window = 9 & 14
Exponentially Weighted MA (EWMA)	$\lambda = 0.9$
Momentum Indicator	Window = 5
Average True Range	Window = 5 & 10
Williams' %R	Window = 14

Table 11: (continued)

Feature Name	Number of Lags/Window Size
Aroon Stochastic Oscillator	Window = 14
Commodity Channel Index	Window = 14
Double Exponential Moving Average	Window = 10
Bollinger Bands (BB_High, BB_Low)	Window = 20
On-Balance Volume	Number of lags = 1
Stochastic Oscillator	Window = 14
Keltner Channels (High, Low, Middle)	Window = 20 & ATR Window = 10
Volume-Weighted Average Price (VWAP)	Window = 14
Volatility Index (VIX) Proxy	ATR Window = 14
Ichimoku Cloud (Tenkan-sen, Kijun-sen, Senkou Span A & B)	Number of lags = 1
Volume-Price Trend (VPT)	Number of lags = 1

C. Hyperparameter Tuning

C1. Forecasting

Within our forecasting analyses, we have merely two models that need hyperparameter tuning; Local Linear Forest and Random Forest. Random Forests are trained using the `Python` package `scikit-learn` and are tuned on the number of trees, the maximum depth of the trees, the minimum split size and the minimum leaf size. Local Linear Forests are tuned equivalently with additional tuning on the λ parameter. The tuning is performed and cross-validated with the `Optuna` package with fifty iterations.

C2. Simulation Study

For our simulation study, all five models are tuned. The Random Forest and Local Linear Forest are tuned similarly as done for forecasting. Gradient boosted trees are implemented from the Python package `xgboost` and are tuned on the number of trees, the maximum depth of the trees and the learning rate. Lasso Random Forest is also implemented using the `scikit-learn` package. It is tuned on the same parameters as Random Forests, with the α from the Lasso regression as an additional parameter. Finally, BART is implemented from the bartpy package and is tuned on the number of chains, the number of trees, the number of samples and how many samples to run without recording. The tuning is performed and cross-validated with the `Optuna` package with fifty iterations for all models, except BART, which only gets twenty iterations due to the high computation time.

D. Additional Simulation Results

As we require forecasts of the same sign as the true value to employ the QLIKE loss function, we can not utilize it for Simulations 2 and 3. Therefore, we rely solely on the RMSE for comparison.

D1. Simulation 2 - Friedman's Example

The results of Simulation 2 can be found in Table 12.

Table 12: Root Mean Squared Error for Simulation 2 (Friedman's Example)

d	n	σ	**LLF**	**RF**	**Lasso-RF**	**BART**	**XGBoost**
10	1000	5	2.133	2.227	**1.791**	1.865	3.376
10	5000	5	1.790	2.091	**1.282**	1.552	3.171
30	1000	5	2.472	2.394	**2.095**	2.461	3.622
30	5000	5	2.012	2.171	**1.365**	1.987	3.177
50	1000	5	2.653	2.470	**2.173**	3.017	3.703
50	5000	5	2.021	2.106	**1.368**	2.213	3.286
10	1000	20	**3.509**	3.704	4.813	4.104	12.257
10	5000	20	**2.399**	2.515	2.903	2.409	12.215
30	1000	20	**4.027**	4.052	4.609	4.648	11.689
30	5000	20	**2.767**	2.855	3.010	3.189	11.970
50	1000	20	**4.103**	4.628	5.453	4.840	11.708
50	5000	20	**2.705**	3.003	3.039	3.531	11.807

Notes: The data ranges over dimensions $d \in \{10, 30, 50\}$, sample size $n \in \{1000, 5000\}$ and error standard deviation $\sigma \in \{5, 20\}$. The lowest RMSEs are highlighted in bold for each case.

Overall, LLF demonstrates competitive performance across varying dimensions, sample sizes, and noise levels. In low noise scenarios, LLF consistently performs well, but Lasso Random Forest clearly has superiority, having the lowest RMSE for all cases where $\sigma = 5$. However, in high noise scenarios ($\sigma = 20$), LLF's performance is particularly noteworthy.

Furthermore, an interesting observation is the performance of XGBoost. It shows inferior performance, across all dimensions and noise levels. This is a clear example that even some modern, complex models can perform very poorly when put in certain situations.

These results show that LLF is a versatile tool in non-parametric regression, providing reliable performance, especially in the presence of significant noise. This demonstrates LLF's strong performance across diverse scenarios, reinforcing its potential for forecasting in complex and volatile environments such as cryptocurrency markets. Its ability to capture intricate patterns and manage high noise levels makes it a promising approach for achieving accurate and reliable volatility prediction.

D2. Simulation 3 - Smoothness and Local Trends

The results, summarized in Table 13, reveal several key insights into the performance of these models under the conditions of smoothness and local trends.

Table 13: Root Mean Squared Error for Simulation 3 (Smoothness and Local Trends)

d	n	σ	**LLF**	**RF**	**Lasso-RF**	**BART**	**XGBoost**
5	1000	0.1	0.251	0.035	**0.034**	0.139	0.074
5	5000	0.1	0.198	0.037	**0.033**	0.217	0.085
20	1000	0.1	0.357	0.025	**0.022**	0.078	0.067
20	5000	0.1	0.275	0.027	**0.022**	0.132	0.067
5	1000	1	**0.196**	0.234	0.246	0.258	0.643
5	5000	1	**0.105**	0.107	0.114	0.158	0.614
20	1000	1	0.309	0.214	**0.183**	0.370	0.618
20	5000	1	**0.126**	0.130	0.138	0.265	0.584
5	1000	2	**0.335**	0.383	0.404	0.375	1.226
5	5000	2	**0.159**	0.180	0.218	0.225	1.178
20	1000	2	**0.283**	0.331	0.382	0.570	1.184
20	5000	2	**0.182**	0.188	0.186	0.390	1.125

Notes: The data ranges over dimensions $d \in \{10, 30, 50\}$, sample size $n \in \{1000, 5000\}$ and error standard deviation $\sigma \in \{5, 20\}$. The lowest RMSEs are highlighted in bold for each case.

Under low noise ($\sigma = 0.1$), Local Linear Forest underperforms, similar to the first two simulations, and especially Lasso Random Forest performs very well. However, as the noise level increases ($\sigma = 1$ and $\sigma = 2$), LLF shows competitive performance and is nearly always the best-performing model. This highlights the robustness of LLF in noisy environments, emphasizing their capability to capture local trends accurately even when the signal is obscured by significant noise. While LLFs do not achieve the lowest RMSE in low-noise settings, their consistent performance across higher noise levels and dimensions highlights their versatility.

E. In-sample Prediction Results

The results of the in-sample predictions can be found in Tables 14, 15, 16. Furthermore, the estimated parameters of GARCH and GJR-GARCH can be found in Tables 17 and 18.

Table 14: In-sample Root-Mean-Squared Errors

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Bitcoin (BTC)	**0.291**	0.362	1.569	1.566	1.709
Ethereum (ETH)	**0.313**	0.506	2.112	2.108	2.155
Tether (USDT)	0.156	**0.148**	0.396	0.396	0.409
Binance Coin (BNB)	**0.297**	0.494	2.086	2.085	2.449
Bitcoin Cash (BCH)	**0.272**	0.639	2.824	2.831	2.766
Litecoin (LTC)	**0.341**	0.721	2.742	2.742	2.890
Internet Computer (ICP)	**0.106**	1.067	4.166	4.164	3.725
Polygon (MATIC)	**0.380**	0.656	3.414	3.414	3.303

Notes: The predictions are made over the entire period of August 1, 2021 - June 1, 2024. The lowest RMSEs are highlighted in bold for each currency.

Interestingly, LLF exhibits the lowest RMSE across all cryptocurrencies, suggesting superior in-sample performance compared to other models. While a lower RMSE typically indicates better model performance, particularly in capturing the variation in observed data, we must tread with caution, as this does not necessarily translate to superior forecasting accuracy in out-of-sample scenarios. LLF's strong performance in-sample may result from overfitting, where the model fits the noise in the data rather than the underlying patterns. Consequently, this could lead to poorer performance when applied to unseen data, diminishing its predictive utility.

In contrast, models such as RF and GARCH variants exhibit higher in-sample RMSE values but may offer more robust and reliable forecasts out-of-sample. Therefore, we must perform an out-of-sample assessment to truly see whether LLF's performance is superior to the other models.

Table 15: In-sample Mean Absolute Errors

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Bitcoin (BTC)	0.230	**0.218**	1.073	1.074	1.182
Ethereum (ETH)	**0.246**	0.309	1.419	1.427	1.452
Tether (USDT)	0.118	**0.067**	0.295	0.295	0.261
Binance Coin (BNB)	**0.233**	0.294	1.334	1.336	1.672
Bitcoin Cash (BCH)	**0.213**	0.435	1.897	1.903	1.772
Litecoin (LTC)	**0.274**	0.379	1.849	1.848	1.963
Internet Computer (ICP)	**0.079**	0.617	3.188	3.188	2.568
Polygon (MATIC)	**0.306**	0.426	2.408	2.408	2.206

Notes: The predictions are made over the entire period of August 1, 2021 - June 1, 2024. The lowest MAEs are highlighted in bold for each currency.

Table 16: In-sample QLIKE

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Bitcoin (BTC)	0.035	**0.007**	0.210	0.210	0.286
Ethereum (ETH)	**0.008**	**0.008**	0.259	0.260	0.273
Tether (USDT)	0.090	**0.018**	0.256	0.256	0.450
Binance Coin (BNB)	0.009	**0.008**	0.257	0.258	0.546
Bitcoin Cash (BCH)	**0.005**	0.011	0.340	0.344	0.279
Litecoin (LTC)	**0.007**	0.008	0.337	0.337	0.424
Internet Computer (ICP)	**0.001**	0.011	0.583	0.583	0.313
Polygon (MATIC)	**0.007**	**0.007**	0.409	0.409	0.326

Notes: The predictions are made over the entire period of August 1, 2021 - June 1, 2024. The lowest QLIKEs are highlighted in bold for each currency.

F. GARCH and GJR-GARCH parameters

Table 17: GARCH Parameter Estimates

	μ	ω	α	β
Bitcoin (BTC)	0.103	1.095	0.127	0.748
Ethereum (ETH)	0.120	0.419	0.083	0.886
Tether (USDT)	-0.009	0.020	0.066	0.841
Binance Coin (BNB)	0.087	0.412	0.157	0.823
Bitcoin Cash (BCH)	-0.171	5.757	0.448	0.367
Litecoin (LTC)	-0.026	0.951	0.115	0.835
Internet Computer (ICP)	-0.130	0.803	0.119	0.858
Polygon (MATIC)	-0.009	0.853	0.132	0.843

Notes: The parameters are estimated using the full data sample (August 1, 2021 - June 1, 2024).

Table 18: GJR-GARCH Parameter Estimates

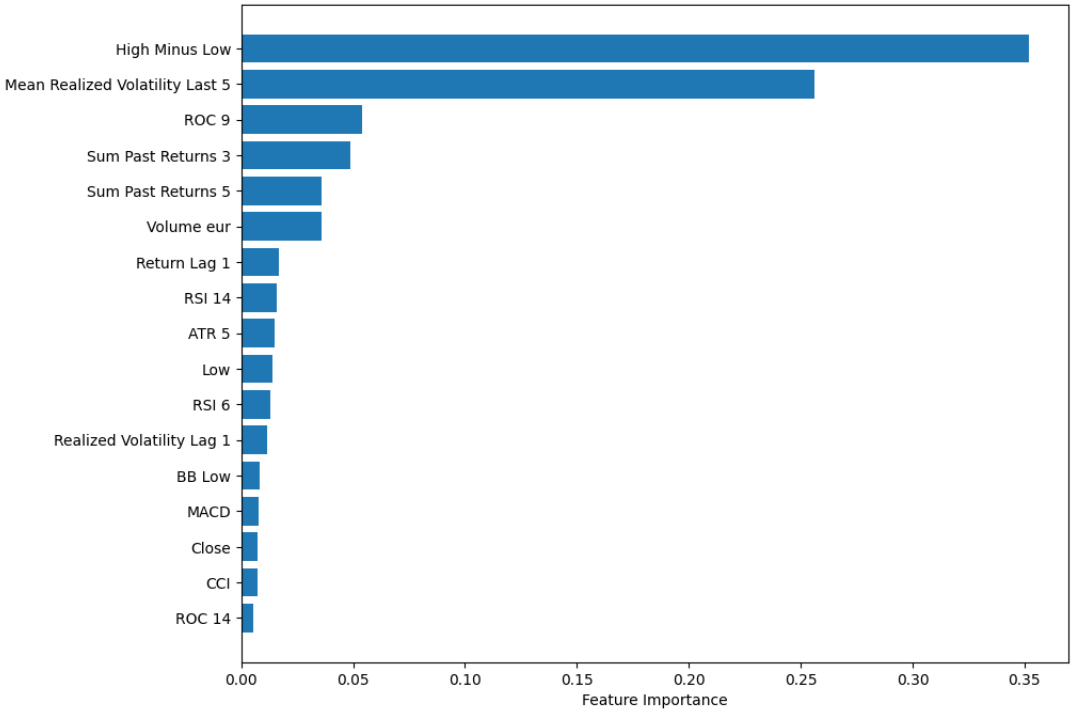
	μ	ω	α	γ	β
Bitcoin (BTC)	0.074	0.969	0.062	0.099	0.777
Ethereum (ETH)	0.116	0.709	0.069	0.063	0.845
Tether (USDT)	-0.008	0.021	0.074	-0.012	0.833
Binance Coin (BNB)	0.076	0.463	0.145	0.040	0.812
Bitcoin Cash (BCH)	-0.07	5.953	0.584	-0.312	0.365
Litecoin (LTC)	-0.002	0.902	0.136	-0.040	0.839
Internet Computer (ICP)	-0.086	0.850	0.144	-0.041	0.853
Polygon (MATIC)	-0.005	0.847	0.134	-0.005	0.843

Notes: The parameters are estimated using the full data sample (August 1, 2021 - June 1, 2024).

G. Feature Importance

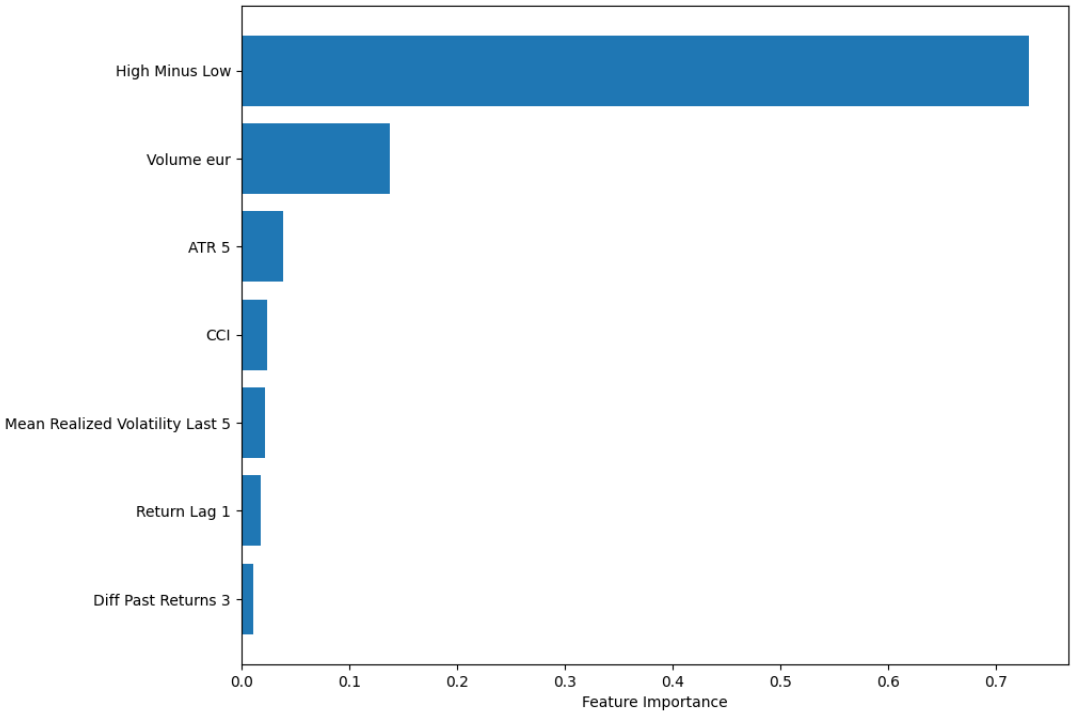
The feature importances for LLF of the remaining cryptocurrencies can be found in Figures 2-8.

Figure 2: Ethereum Feature Importance for Local Linear Forests



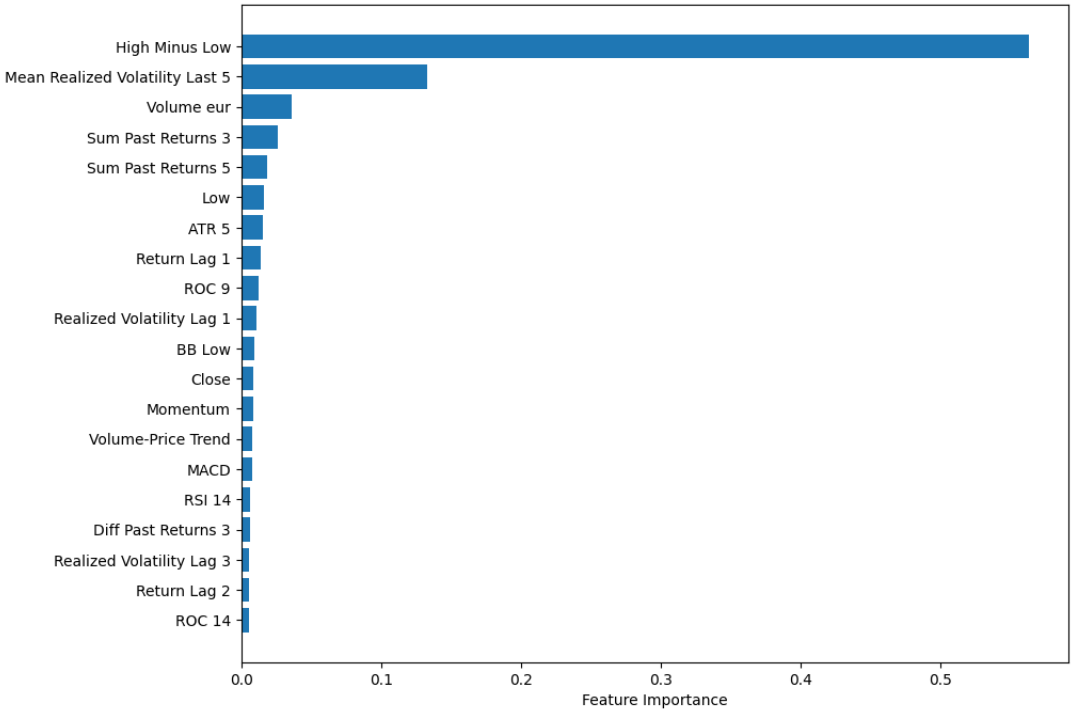
Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

Figure 3: Tether Feature Importance for Local Linear Forests



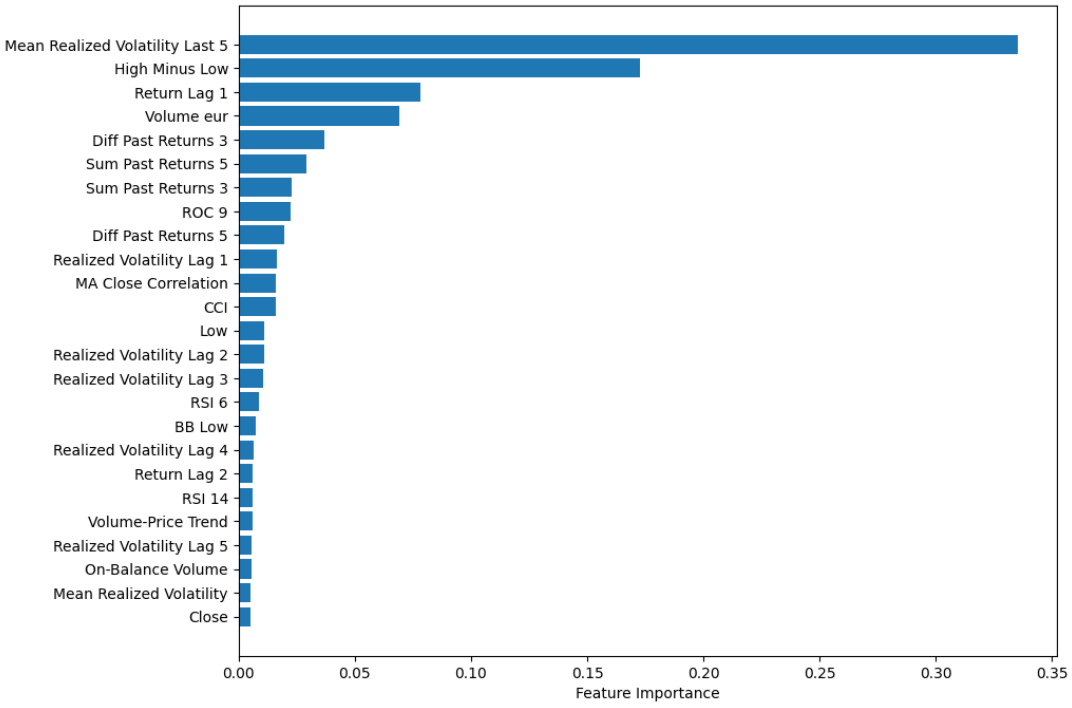
Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

Figure 4: Binance Coin Feature Importance for Local Linear Forests



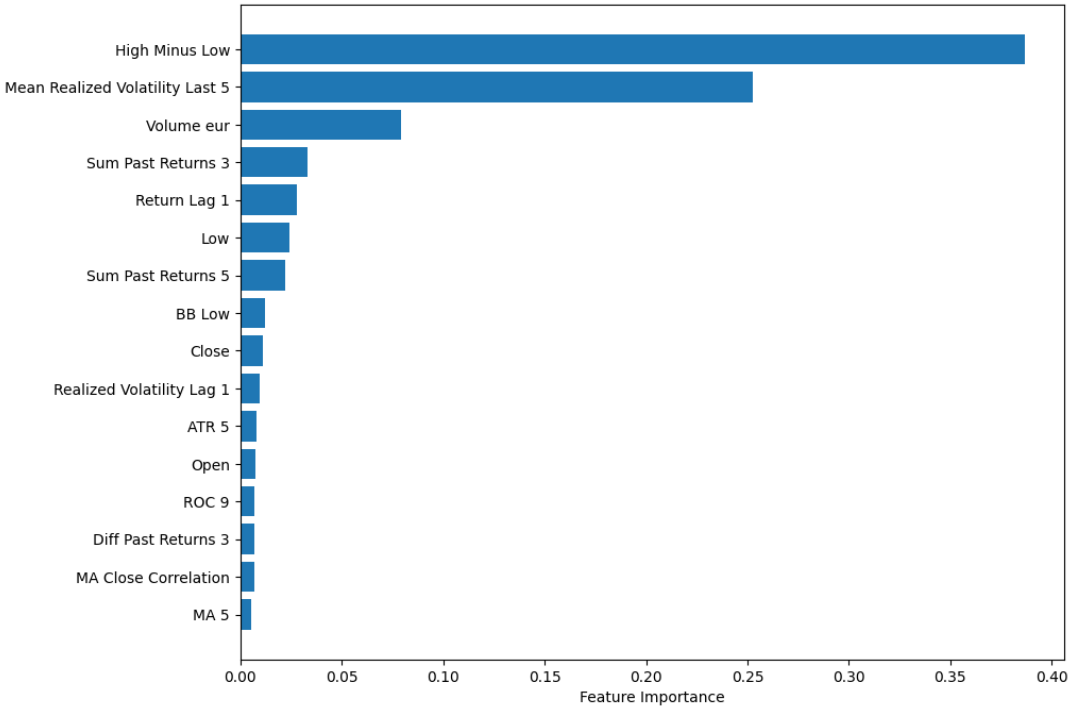
Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

Figure 5: Bitcoin Cash Feature Importance for Local Linear Forests



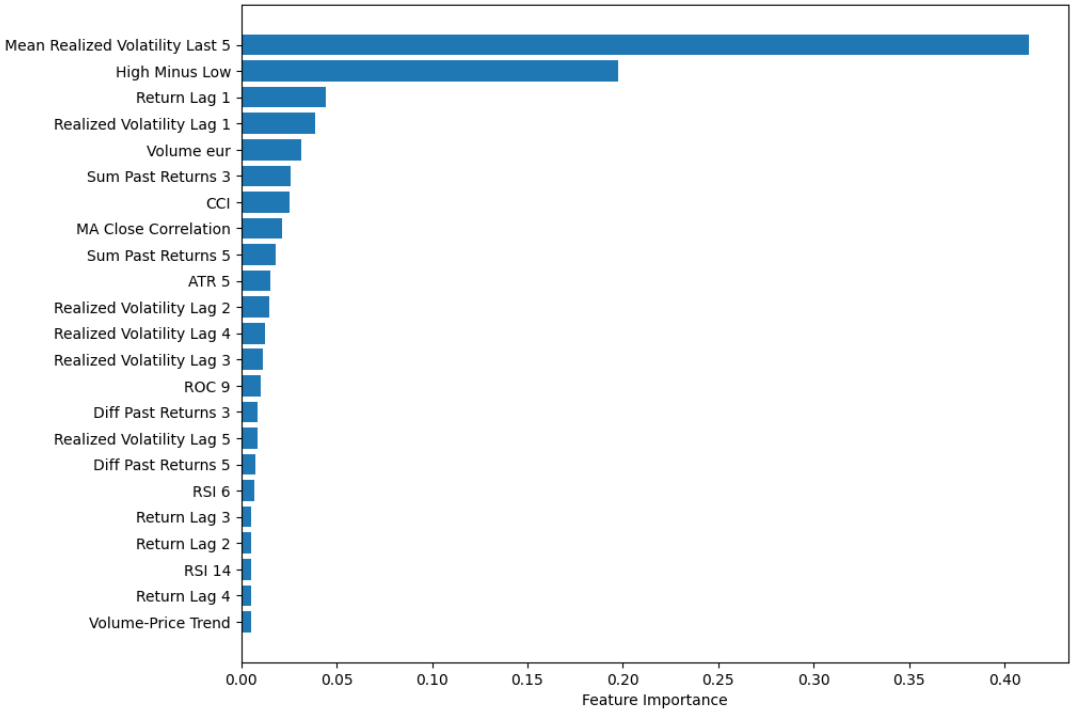
Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

Figure 6: Litecoin Feature Importance for Local Linear Forests



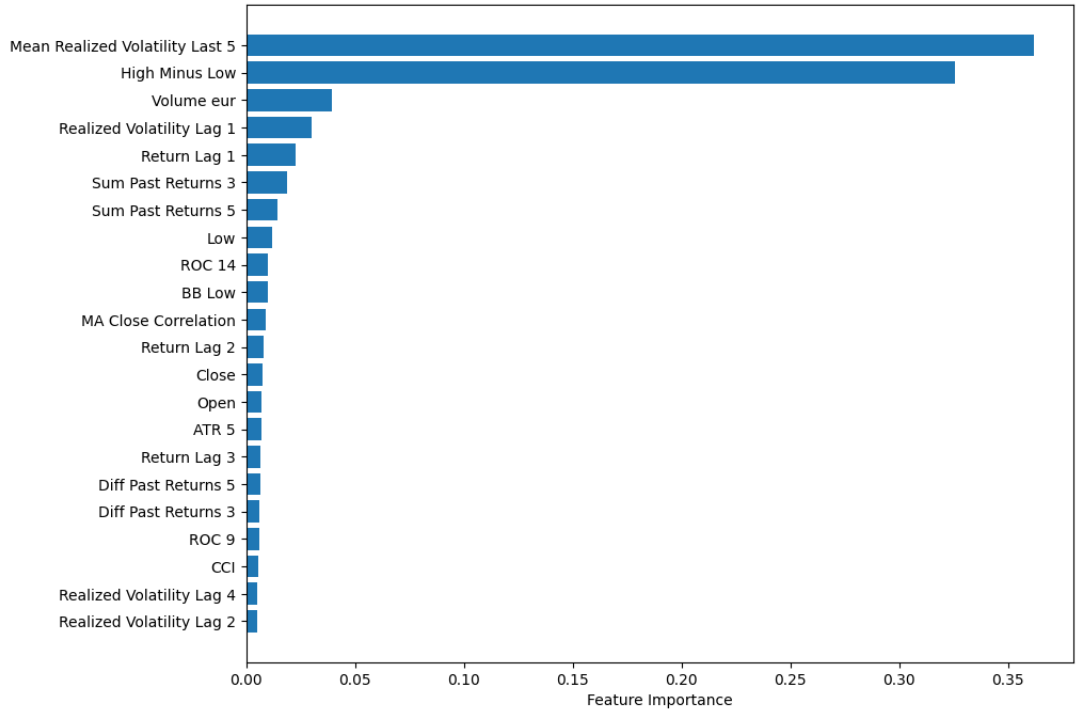
Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

Figure 7: Internet Computer Feature Importance for Local Linear Forests



Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

Figure 8: Polygon Feature Importance for Local Linear Forests



Notes: For brevity and clarity, only the features are included with feature importance greater than 0.5%.

H. Market Cycles

Table 19: Number of Observations per Market Cycle per Coin

	Bull	Consolidating	Bear
Bitcoin (BTC)	313	524	220
Ethereum (ETH)	319	468	270
Tether (USDT)	0	1057	0
Binance Coin (BNB)	297	525	235
Bitcoin Cash (BCH)	265	443	349
Litecoin (LTC)	292	457	308
Internet Computer (ICP)	280	327	450
Polygon (MATIC)	317	341	399

Notes: The market cycles are determined based on a moving window of thirty days. If the prices increase by 5% or more in thirty days, it is considered a bullish phase. If they decrease by 5% or more, it is classified as a bearish phase. The remainder is categorized as the consolidating phase.

As illustrated in Table 19, Tether (USDT) presents a distinct pattern in market behaviour compared to other cryptocurrencies analyzed, as it does not exhibit traditional market cycles. This unique characteristic stems from Tether's fundamental nature as a stablecoin. The absence of observations in the bullish and bearish phases reflects its stable valuation and its exclusion from typical market cycle analyses. Therefore, from this point onwards in our analysis, Tether is intentionally excluded from further consideration in the market cycle analysis. This decision

acknowledges Tether's unique role as a stablecoin and recognizes that its stable valuation stands apart from the speculative price dynamics observed in other cryptocurrencies. For the remaining cryptocurrencies, the results of the market cycle analysis can be found in Tables 21, 22, 23, 24, 25 and 26. Furthermore, we evaluate LLF's performance using the MCSR in each phase separately, based on the RMSE, as shown in Table 20.

Table 20: Model Confidence Set Rates Across Market Phases

	LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Bear	**1.000**	0.571	0.000	0.000	0.000
Consolidating	**1.000**	0.429	0.000	0.000	0.000
Bull	**1.000**	0.571	0.000	0.000	0.000

Notes: MCSRs at $\alpha = 0.1$ are noted for each of the market phases, based on the RMSE. The highest MCSRs are highlighted in bold for each currency.

We observe that Local Linear Forests always belong to the superior models for each cryptocurrency (excluding Tether), across all phases in the market cycle. Random Forests are part of the MCS about half of the time and therefore LLF is the only consistently good choice. Just like in our other analyses, the GARCH-type and HAR-RV models are unable to match the performance of the ensemble methods.

Table 21: Statistical Summary of Squared Forecast Errors Across Market Cycles for Ethereum

		LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Mean *(St. Dev.)*	Bear	**0.056**	0.315	2.599	2.691	2.639
		(0.08)	*(0.53)*	*(4.85)*	*(4.79)*	*(5.64)*
	Consolidating	**0.057**	0.227	1.203	1.216	0.789
		(0.08)	*(0.47)*	*(1.46)*	*(1.35)*	*(1.33)*
	Bull	**0.085**	1.115	3.336	3.250	3.250
		(0.20)	*(5.08)*	*(10.84)*	*(11.29)*	*(11.05)*
Kruskal-Wallis	H-statistic	2.27	11.09	3.32	4.67	12.50
	P-value	0.32	0.004*	0.190	0.097	0.002*
Dunn	Bear-Bull	0.673	0.645	0.954	0.122	1.000
	Bear-Consolidating	1.000	0.618	0.226	0.152	0.009*
	Bull-Consolidating	0.533	0.003*	1.000	1.000	0.020*

Notes: For Dunn's test, the test statistic is noted. Significance at the $\alpha = 0.05$ level is indicated by an asterisk (*). The lowest mean squared forecast errors are highlighted in bold.

Table 22: Statistical Summary of Squared Forecast Errors Across Market Cycles for Binance Coin

		LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Mean *(St. Dev.)*	Bear	**0.441**	0.601	4.470	4.733	3.945
		(1.39)	*(2.29)*	*(12.44)*	*(12.44)*	*(11.75)*
	Consolidating	**0.289**	0.291	1.325	1.466	1.261
		(0.82)	*(0.68)*	*(3.30)*	*(3.45)*	*(3.38)*
	Bull	**0.737**	1.001	3.025	2.957	3.161
		(1.59)	*(3.59)*	*(10.32)*	*(10.48)*	*(10.92)*
Kruskal-Wallis	H-statistic	14.52	7.48	6.52	8.455	12.57
	P-value	7.0e-04*	0.024*	0.038*	0.015*	0.002*
Dunn	Bear-Bull	0.486	0.127	0.793	0.078	1.000
	Bear-Consolidating	1.000	1.000	0.089	0.012*	0.256
	Bull-Consolidating	4.2e-04*	0.047*	0.227	0.959	0.002*

Notes: For Dunn's test, the test statistic is noted. Significance at the $\alpha = 0.05$ level is indicated by an asterisk (*). The lowest mean squared forecast errors are highlighted in bold.

Table 23: Statistical Summary of Squared Forecast Errors Across Market Cycles for Bitcoin Cash

		LLF	**RF**	**GARCH (1,1)**	**GJR-GARCH**	**HAR-RV**
Mean *(St. Dev.)*	Bear	**0.530**	0.882	2.941	3.019	3.384
		(0.64)	*(1.70)*	*(4.665)*	*(4.93)*	*(9.00)*
	Consolidating	**0.419**	1.027	3.435	3.568	3.348
		(0.64)	*(2.75)*	*(7.20)*	*(7.58)*	*(8.87)*
	Bull	**0.735**	2.504	17.429	21.399	7.489
		(1.37)	*(6.89)*	*(87.96)*	*(119.79)*	*(21.50)*
Kruskal-Wallis	H-statistic	5.41	21.37	3.89	3.23	20.76
	P-value	0.066	2.3e-05*	0.143	0.198	3.1e-05*
Dunn	Bear-Bull	1.000	0.065	0.504	0.668	8.2e-04*
	Bear-Consolidating	0.341	0.226	1.000	1.000	1.000
	Bull-Consolidating	0.082	1.1e-05*	0.175	0.246	1.1e-04*

Notes: For Dunn's test, the test statistic is noted. Significance at the $\alpha = 0.05$ level is indicated by an asterisk (*). The lowest mean squared forecast errors are highlighted in bold.

Table 24: Statistical Summary of Squared Forecast Errors Across Market Cycles for Litecoin

		LLF	RF	GARCH (1,1)	GJR-GARCH	HAR-RV
Mean *(St. Dev.)*	Bear	**0.449**	1.128	5.593	5.621	6.172
		(1.07)	*(4.48)*	*(15.04)*	*(15.03)*	*(21.22)*
	Consolidating	**0.241**	0.276	1.829	1.833	1.316
		(0.41)	*(0.47)*	*(1.59)*	*(1.59)*	*(2.01)*
	Bull	**0.622**	1.874	5.239	5.303	5.986
		(0.91)	*(6.10)*	*(14.91)*	*(14.54)*	*(12.32)*
Kruskal-Wallis	H-statistic	10.16	12.51	12.86	12.79	17.42
	P-value	0.006*	0.002*	0.002*	0.002*	1.6e-04*
Dunn	Bear-Bull	0.412	0.029*	0.062	0.105	1.000
	Bear-Consolidating	0.354	1.000	0.001*	0.001*	0.002*
	Bull-Consolidating	0.006*	0.001*	1.000	1.000	0.002*

Notes: For Dunn's test, the test statistic is noted. Significance at the $\alpha = 0.05$ level is indicated by an asterisk (*). The lowest mean squared forecast errors are highlighted in bold.

Table 25: Statistical Summary of Squared Forecast Errors Across Market Cycles for Internet Computer

		LLF	RF	GARCH (1,1)	GJR-GARCH	HAR-RV
Mean *(St. Dev.)*	Bear	**0.424**	1.550	4.218	4.386	4.115
		(1.03)	*(3.00)*	*(9.46)*	*(9.78)*	*(9.30)*
	Consolidating	**0.400**	1.304	2.321	2.394	2.276
		(1.04)	*(3.37)*	*(5.74)*	*(5.79)*	*(6.05)*
	Bull	**0.632**	6.629	12.269	12.765	11.653
		(1.46)	*(27.14)*	*(55.11)*	*(54.59)*	*(46.84)*
Kruskal-Wallis	H-statistic	4.75	8.21	9.44	7.98	14.87
	P-value	0.093	0.016*	0.009*	0.018*	5.9e-04*
Dunn	Bear-Bull	1.000	0.182	0.333	0.500	0.175
	Bear-Consolidating	0.745	1.000	0.617	0.635	0.267
	Bull-Consolidating	0.088	0.016*	0.007*	0.014*	3.5e-04*

Notes: For Dunn's test, the test statistic is noted. Significance at the $\alpha = 0.05$ level is indicated by an asterisk (*). The lowest mean squared forecast errors are highlighted in bold.

Table 26: Statistical Summary of Squared Forecast Errors Across Market Cycles for Polygon

		LLF	RF	GARCH (1,1)	GJR-GARCH	HAR-RV
Mean ***(St. Dev.)***	Bear	**0.623**	0.753	3.444	3.462	3.360
		(1.35)	*(1.79)*	*(7.50)*	*(7.64)*	*(8.96)*
	Consolidating	0.526	**0.499**	2.830	2.648	1.823
		(1.08)	*(1.15)*	*(3.78)*	*(3.77)*	*(3.81)*
	Bull	**0.851**	1.335	3.282	3.256	4.194
		(1.74)	*(4.34)*	*(12.28)*	*(12.15)*	*(11.02)*
Kruskal-Wallis	H-statistic	5.44	13.91	5.86	6.06	9.25
	P-value	0.066	9.5e-04*	0.053	0.048*	0.009*
Dunn	Bear-Bull	0.532	0.359	0.089	0.077	1.000
	Bear-Consolidating	0.962	0.086	1.000	1.000	0.045*
	Bull-Consolidating	0.059	6.9e-04*	0.109	0.105*	0.019*

Notes: For Dunn's test, the test statistic is noted. Significance at the $\alpha = 0.05$ level is indicated by an asterisk (*). The lowest mean squared forecast errors are highlighted in bold.

I. Code Description

Here, a brief description of the code's structure is given to enhance reproducibility of the results.

First, all of the data files can be found within the 'data' folder. In the 'forecasting' folder, specifically in the 'forecasts' subfolder, all of the results can be obtained by running 'perform_forecasts.ipynb'. It first obtains the data using the 'obtain_data.py', which are found within 'data_preprocessing'. It forecasts volatility using the 'in_sample_forecasts.py' and 'out_of_sample_forecasts.py' within the 'forecasts' folder. Additionally, the 'forecasting' subfolder contains files for the models, hyperparameter tuning of those models and several files in 'utils' for calculating errors, utility, Model Confidence Sets and feature importance, which are all imported into the forecasts.

Lastly, the 'simulation_study' folder contains two files that run the three simulations, as simulations 2 and 3 are contained within one file. Furthermore, the folder contains files for hyperparameter tuning and the models, which are imported into the simulations.