Tree-Based Methods of Volatility Prediction for the S&P 500 Index

Marin Lolic
Independent Researcher
Baltimore, MD, USA
Marin.Lolic@gmail.com

Abstract

Predicting asset return volatility is one of the central problems in quantitative finance. These predictions are used for portfolio construction, calculation of value-at-risk (VaR) and pricing derivatives such as options. Classical methods of volatility prediction utilize historical returns data and include exponentially weighted moving average (EWMA) and generalized autoregressive conditional heteroskedasticity (GARCH). These approaches have shown significantly higher rates of predictive accuracy than corresponding methods of return forecasting, but they still have vast room for improvement. In this paper, we propose and test several methods of volatility forecasting on the S&P 500 Index using tree ensembles from machine learning, namely random forest and gradient boosting. We show that these methods generally outperform the classical approaches across a variety of metrics on out-of-sample data. Finally, we use the unique properties of tree-based ensembles to assess what data can be particularly useful in predicting asset return volatility.

Keywords: Volatility, Tree Ensembles, GARCH, Prediction, Machine Learning

1. Introduction

Finance acquired its most elemental measurements from statistics, including metrics such as mean, standard deviation and correlation. Mean became the measure of average returns over some period of time, and if applied to the future, expected return. Standard deviation, or its squared analog, variance, came to measure the dispersion of returns around the mean (Ang 2014). This metric took on the name volatility, and it holds a crucial place in much of modern finance. Equation 1 defines sample standard deviation, or volatility, where r_t is a given realized return from a time series, while $\bar{r}$ is the sample mean of the series.

$$s = \sqrt{\frac{\sum_{t=1}^{n}(r_t - \bar{r})^2}{n-1}} \tag{1}$$

While higher expected return is almost always desired in an asset or a portfolio, higher volatility has usually become something to avoid - indeed, the standard mean-variance optimization framework explicitly penalizes variance in a portfolio by equating it with risk (Cornuejols et al. 2018). Volatility is a central component of most risk management systems and is related to tail risk measures like value-at-risk and expected shortfall. Pricing options and many other derivatives also necessitates volatility, as originally described by Black and Scholes (1973). In these and other applications, what matters is typically not historical volatility but its future realization. Fortunately, financial academics and practitioners have long known that future volatility is more persistent, and thus more predictable, than future returns (Mandelbrot 1963). To take advantage of this relative predictability, there are a number of "classical" methods, such as simple moving averages, exponentially weighted moving averages, and the ARCH/GARCH family of models. For assets with liquid options, one could also calculate the implied volatility of the options from market prices, though this does not answer how market participants set the options prices in the first place.

Our goal is to improve on the existing methods by introducing the use of tree-based machine learning ensembles such as random forest and gradient boosting. These non-parametric methods developed outside of finance and economics but have achieved impressive results when applied to forecasting in a variety of fields. More specifically, we use recent daily returns to predict 21-day ahead volatility, which corresponds to one month's worth of trading days. We show that tree ensembles, particularly gradient boosting, can produce superior volatility forecasts on the S&P 500, one of the most popular stock indices in the United States. We also show that even better results can be obtained from combining historical returns data with a market-based volatility estimate in the form of the CBOE VIX Index. All of these results are calculated on out-of-sample test data, temporally separated from the training data used to build the models. Finally, we look "under the hood" of our random forest and gradient boosting models to understand the time lags that are especially significant for volatility forecasts. We find strong evidence for exponentially decaying relevance of older data, similar to EWMA. The improved accuracy of tree-based methods therefore likely results from their ability to process data in a non-linear way.

2. Background

2.1 Volatility Prediction

Simple methods of volatility forecasting include trailing moving averages of recent volatility, e.g., using realized volatility over the past *x* days as a prediction of volatility over the next *y* days. This assumes that the near future will resemble the near past, an often-reasonable assumption given the tendency of volatility to cluster across time. A more sophisticated, and often more accurate, approach is exponentially weighted moving average (EWMA), which gives more recent observations a greater weight compared to older observations (Miller 2019). Equation 2 shows the EWMA predicted variance as a function of realized returns (r_t), the mean of the returns (μ_t, sometimes assumed to be 0), and a decay factor (δ) that ranges between 0 and 1.

$$\sigma_t^2 = A\sum_{i=0}^{n-1}\delta^i r_{t-i}^2 - B\mu_t^2$$
$$A = \frac{S_1}{S_1^2 - S_2}$$
$$B = S_1 A$$
$$S_1 = \frac{1-\delta^n}{1-\delta}$$
$$S_2 = \frac{1-\delta^{2n}}{1-\delta^2} \tag{2}$$

The field of econometrics advanced the prediction of volatility with the introduction of the autoregressive conditional heteroskedasticity (ARCH) model by Engle (1982). ARCH forecasts future variance based on long-run variance and recent disturbances. Bollerslev (1986) extended the framework to include lags of previous predictions, creating the popular generalized autoregressive conditional heteroskedasticity (GARCH) model, whose 1-lag version is seen in Equation 3. The model estimates three coefficients that should sum to one: α_0 for long-run variance, α_1 for lagged values of the residual, and β for lagged values of predicted variance. GARCH allows for additional lags of either the autoregressive term or the residual term or both, though the functional form is always additive. Later researchers created a vast number of additional refinements to GARCH, such as exponential (EGARCH), integrated (IGARCH), threshold

(TGARCH) and others (for a typical comparison, see Lim and Sek 2013). Most modern commercial software packages for forecasting volatility use either a variant of GARCH or EWMA.

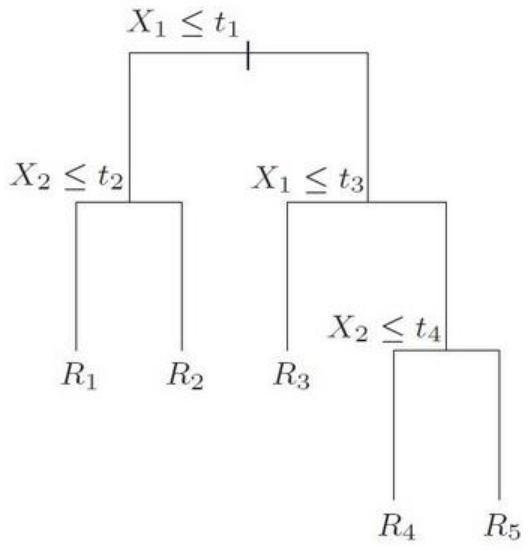
$$\sigma_t^2 = \alpha_0\bar{\sigma}^2 + \alpha_1\sigma_{t-1}^2 u_{t-1}^2 + \beta\sigma_{t-1}^2 \quad \textbf{(3)}$$

In recent years, academics have published numerous new approaches to volatility prediction. Corsi (2009) introduced the heterogeneous autoregressive (HAR) model, which posits that a linear cascade of daily, weekly and monthly values of realized volatility can predict future volatility. As part of the deep learning boom of the last decade, models like those of Bucci (2020) have utilized various neural network architectures for financial market forecasting. The new approaches also sometimes incorporate data beyond historical returns such as internet search frequency (Xiong et al. 2015). While deep learning approaches have reported strong results in volatility prediction, they suffer from the "black box" problem of limited insight into how and why they work. Still other research has used intraday returns as inputs, which is generally not feasible for the classical models (Andersen et al. 2003).

2.2 Tree-Based Machine Learning

Machine learning originated in the mid-twentieth century at the intersection of the fields of statistics and computer science. Its goal was the development of algorithms that could "learn" patterns in training data, often for purposes of prediction on previously unseen data. Decades of research have produced many approaches, including penalized linear models, support vector machines, neural networks, and more (see James et al. 2021 for an introduction). An important strand of machine learning research started with the introduction of classification and regression trees (CART) by Breiman et al. (1984). CART builds a binary tree where each split is a certain feature of the data at a certain cutoff point, as shown in Figure 1 below. The algorithm chooses the features and cutoff points by minimizing a loss function, such as mean squared error for regression and Gini impurity for classification. This means CART is a greedy algorithm, making the locally optimal choice at each stage. The added step of pruning the tree by deleting nodes with small sample sizes can reduce overfitting.

Figure 1. Diagram of CART Construction
X_1 and X_2 are data features, while t_1 through t_4 are cutoff values



Source: Hastie et al. (2009)

While individual trees offer a highly tractable method of machine learning, they typically perform poorly compared to other methods. To remedy this defect, Breiman introduced the idea of tree ensembles, or models composed of many trees. In bootstrap aggregation, or bagging, a preset number of trees are built on bootstrapped versions of the original dataset (Breiman 1996). Though each tree may have a high error rate, the aggregate of all the predictions is typically much more accurate. Breiman (2001) further improved on bagging by selecting features at each split point from a random subset of all features, known as a random forest. Despite being somewhat counterintuitive, this method combats overfitting the training data by occasionally forcing the model to use features it may otherwise ignore. The model builder must set the number of features to sample from at each split point, a tunable hyperparameter often denoted as *mtry*. Other tunable hyperparameters in typical random forest models include maximum tree depth, minimal sample size for a node, the exact splitting rule to use, etc.

Random forest builds numerous trees in parallel, but the gradient boosting approach proposed by Friedman (2002) builds trees in sequence. While the first tree is fit to the observed data, each subsequent tree is fit to the residuals of the prior tree. Often, the individual trees only consist of the root and two nodes, and the model's predictive power comes entirely from the sequential construction of trees. Much like random forests, gradient boosting requires several tunable hyperparameters such as tree depth, learning rate, and the total number of trees. When done correctly, this can prevent overfitting by pushing the model to learn more general patterns instead of simply memorizing the training data. As with most methods of machine learning, tuning is typically done with n-fold cross-validation, while the final model is evaluated on a test set of data it has not seen before.

3. Materials and Methods

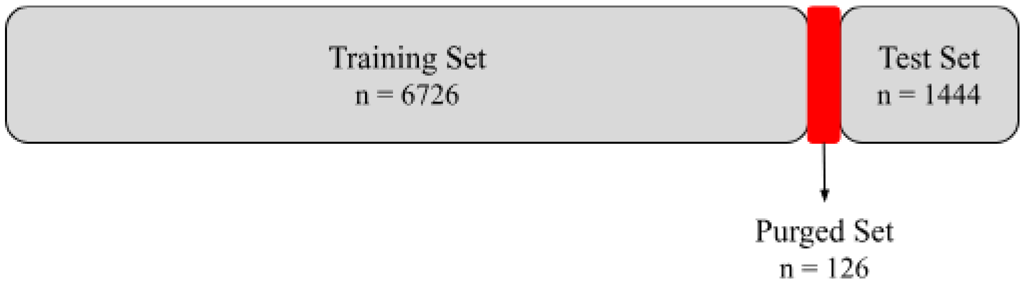
3.1. Preliminary Decisions

To offer a fair comparison between classical methods and our approach, we limit ourselves to daily returns as opposed to using intraday ones. More specifically, the daily returns are simple returns, not logarithmic. Instead of calculating new features from our data such as the trailing volatilities in the HAR model, we leave the daily returns as the predictors; this will become significant when we analyze feature importance. We employ the following conventions for trading days: 21 trading days in a month, 93 in a quarter, 126 in six months, and 252 in a full year. Finally, all standard deviation statistics will be reported as annualized numbers, which will necessitate scaling by the square root of time.

3.2. Data and Preprocessing

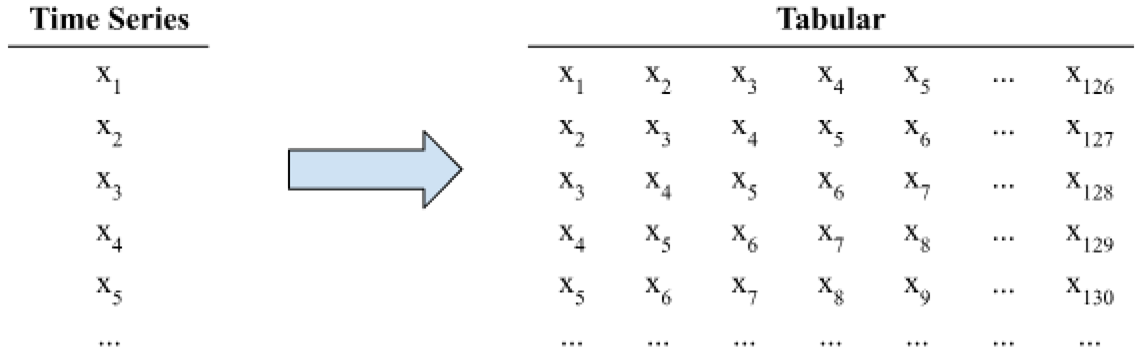
Our data consists of 31 years of daily returns for the S&P 500 Index, a very popular capitalization-weighted index of stocks traded in the United States; our full time series begins in February 1993 and ends in March 2024. While the S&P 500 has historical data starting in the 1950s, we restrict the sample to a more recent period due to the possibility that older data is less relevant to modern markets. We partition our 7846 data points into three non-overlapping pieces - a training set, a purged set, and a test set - as depicted in Figure 2 below. While the training set and test set are standard in machine learning, the purged set is somewhat unusual and follows the recommendation of Lopez de Prado (2018). In time series data like ours, serial correlation can lead to informational leakage when the test data directly follows the training data. By "purging", or discarding, data between the two sets, we can reduce this risk and protect the validity of our conclusions. We purge 126 data points, or six months of daily data, for reasons we will soon explain.

Figure 2. Data Partitioning



Tree-based methods generally accept data in tabular form, with rows corresponding to individual samples and columns corresponding to features of the data. To accommodate the format, we take "slices" (also called windows) of our data in both the training and test sets, with each slice taking up one row and consisting of 126 consecutive daily returns. This means our features are the daily returns themselves, one for each trading day in a six-month period. There is nothing special about 126 days of trading data, but it should be long enough to capture autocorrelation in volatility without being so long that older data is irrelevant. Consecutive slices share 124 of 126 data points, as using non-overlapping slices would lead to a sample size of only 62. While the 126 daily returns serve as our features, our target for prediction is the standard deviation of returns for the subsequent 21 days. Figure 3 provides a visual depiction of the slicing process, which we use on both the training and test sets. We purged exactly 126 data points between our training and test sets because that is the length of one of our data slices.

Figure 3. Slicing Time Series Data Into Tabular Form



x_1	x_2	x_3	x_4	x_5	...	x_{126}
x_2	x_3	x_4	x_5	x_6	...	x_{127}
x_3	x_4	x_5	x_6	x_7	...	x_{128}
x_4	x_5	x_6	x_7	x_8	...	x_{129}
x_5	x_6	x_7	x_8	x_9	...	x_{130}
...	...	...	...	...	...	...

3.3. Prediction Methods

Using the R programming language and associated libraries, we examine seven total methods of volatility prediction - three classical, one using options, and three based on decision trees. The first method is an equal-weighted moving average of past values. This simply involves calculating the realized volatility of the final 21 days of each data slice and using it as the prediction for the following 21 days; it implicitly assumes that each day in the past contributes an equal amount of information. The second method utilizes an exponentially-weighted moving average. The EWMA uses all 126 available data points in each data slice, but it weights the more recent data points more heavily. We tune the important decay factor (δ) hyperparameter with 10-fold cross-validation on the training data, using mean squared error (MSE) as the value to minimize. The final classical prediction approach is a GARCH (1,1) model. As GARCH generally requires more than 100 data points to specify, we use the full training data time series instead of relying on slices. Our options-based prediction method is based on the CBOE VIX Index, which tracks implied volatility on at-the-money one-month options on the S&P 500 Index. For each of the 21-day prediction periods, our volatility forecast is simply the closing value of the VIX on the day before the prediction period started.

The first decision tree method is random forest, for which we employ 500 trees without a limit on tree depth. The sole hyperparameter to tune through 10-fold cross-validation is the number of features to sample at each split point; we select this value by performing a grid search with options ranging from 10 to 120 in increments of 10. Intuitively, this corresponds to either using a small subset of historical data (10 days out of 126 possible) or a large subset, or some intermediate value. The lack of limit on tree depth makes this a computationally intensive model, so we use 20 parallel CPU cores to speed up the calculations. The second decision tree method is XGBoost, a variant of gradient boosting. We tune two hyperparameters using 10-fold cross-validation: the learning rate of the model and the total number of trees; the learning rate varies from 0.05 to 0.30 in increments of 0.05, while the number of trees ranges from 50 to 500 in steps of 50. We set maximum tree depth to one, though we could also tune this as well. Our final method is also gradient boosting using XGBoost, but with the addition of the VIX value as another predictor in addition to the 126 historical daily returns. While there are many ways to combine forecasts, we favor this approach because it allows the gradient boosting model to determine the relative importance of VIX versus the historical data. Our tuning procedure for gradient boosting + VIX is the same as for regular gradient boosting.

3.4. Evaluation Metrics

Our goal throughout this paper is to predict future volatility (standard deviation) of returns, a continuous and real-valued quantity. We will use three metrics to evaluate the performance of all the models tested, the first of which is root mean squared error (RMSE), seen in Equation 4. RMSE is a very common measure of success for regression tasks, but we will supplement it with mean absolute error (MAE), as shown in Equation 5. MAE is similarly concerned with average errors, but its use of absolute value reduces the impact of outliers compared to RMSE. Finally, we will use mean absolute percentage error (MAPE), a scaled metric defined in Equation 6. In equations 4–6, y_i is the true value of volatility, while $\hat{y}_i$ is the predicted value.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{4}$$

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n} \tag{5}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{6}$$

4. Results

4.1. Forecast Comparisons

Table 1 below shows the performance of each of the seven methods across the three metrics; all numbers are calculated on the out-of-sample test set, and lower values are always better. One result is that none of the classical methods clearly dominate the others. Simple moving average is tied for the worst RMSE but has a lower MAE than GARCH. EWMA improves on simple moving average across all metrics, but the improvement is generally small. GARCH has high RMSE and MAE but lower MAPE than the moving averages; this is because it is more likely to underestimate volatility, and MAPE penalizes underestimation less than overestimation. Using VIX, which requires no modeling or calculations, results in a lower RMSE than the classical methods, an average MAE, and the worst MAPE among all models. We believe this is because VIX is more successful at predicting outlier values of volatility, especially very high values during periods of market stress. However, its average predictions are higher than those of other models due to the volatility risk premium in options (Ilmanen 2011), resulting in a poor MAPE. The random forest clearly beats the classical methods and VIX, while gradient boosting comes ahead of random forest on RMSE and MAPE. Overall, however, the best model across all metrics is the final one - gradient boosting with the addition of VIX as another predictor.

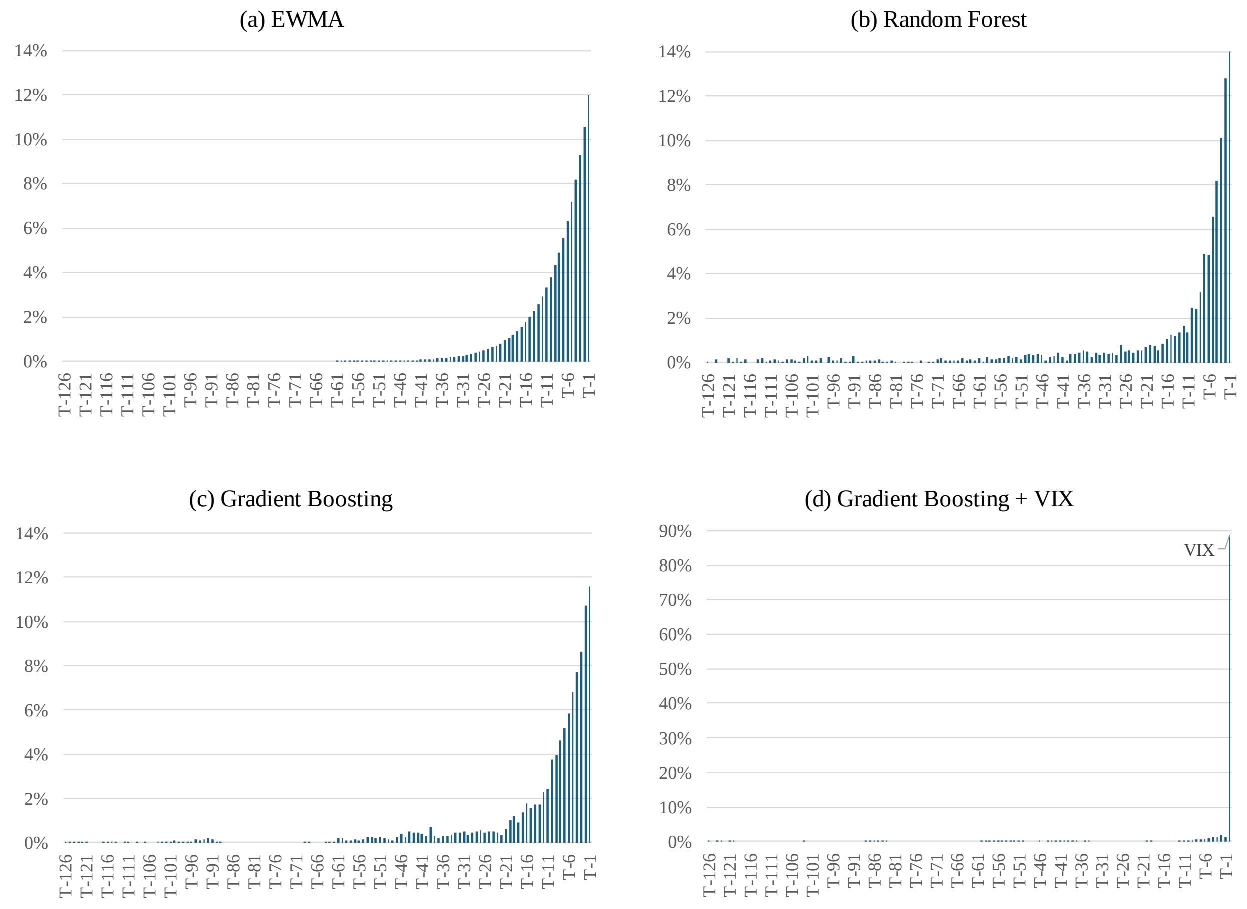
Table 1. Comparison of Forecast Methods
Bolded Values Indicate Lowest (Best)

Method	Hyperparameters	RMSE	MAE	MAPE
Simple Moving Average	Lookback = 21 days	12.3%	6.7%	38.3%
EWMA	$\delta = 0.88$	11.5%	6.5%	36.4%
GARCH	(p, q) = (1, 1)	12.2%	7.2%	32.4%
VIX		10.6%	6.9%	48.0%
Random Forest	mtry = 30, trees = 500	9.9%	5.2%	31.5%
Gradient Boosting	eta = 0.1, trees = 450	9.6%	5.3%	31.1%
Gradient Boosting + VIX	eta = 0.1, trees = 250	**9.5%**	**5.0%**	**29.0%**

4.2. Variable Importance

Tree-based methods of machine learning allow a degree of insight into how the model makes its predictions. For random forests and gradient boosting, we can answer this question through variable importance analysis. Each time a tree is split, the variable importance algorithm tracks the splitting predictor as well as how much the sum of squared errors (SSE) is reduced. By averaging across all trees created, variable importance produces a relative ranking of all the predictors. Figure 4 illustrates the variable importance measures for our three tree-based models and compares them to EWMA weights.

Figure 4. Variable Importance



In the random forest and both versions of gradient boosting, recent returns are far more predictive of future volatility than returns at the start of the 126-day period. The decline in variable importance as returns recede further into the past is largely consistent with exponential decay, possibly explaining why EWMA is a popular choice for volatility prediction. However, both our random forest model and the gradient boosting model (without VIX) meaningfully outperform EWMA, suggesting that the tree ensembles add predictive power. We think this is due to the non-linear nature of tree ensembles; while EWMA and GARCH are confined to relatively simple, additive representations of volatility, deep and independent trees (as in random forest) or small sequential trees (as in gradient boosting) can capture more complex patterns in historical returns data. In addition, the binary nature of regression trees allows the models to learn differences between historical returns above the mean and those below the mean. Finally, in the gradient boosting model including VIX, we see that VIX itself is the dominant explanatory variable. While the returns data collectively contribute only 12% of the total SSE reduction in the model, the results are far better than VIX alone across all three evaluation metrics.

5. Discussion and Concluding Remarks

We believe our work is relevant to both academics and practitioners. By demonstrating the value of tree ensembles in this field, we may open new avenues for understanding asset behavior while also providing a practical recipe to traders and investment managers who rely on volatility predictions. But enhanced predictive power is not the only virtue of the models we have built. They can also accommodate external explanatory variables such as VIX, while EWMA and GARCH generally do not. A further benefit of our approach is its potential flexibility. Different models can be trained for various asset classes or factors, but also for different volatility prediction time horizons. Many financial risk management systems calculate volatility at one time horizon and then simply scale it to other horizons. Unfortunately, this is only valid if volatility is constant across the time period, which is typically not the case in reality. While building and maintaining a stable of tree-based models requires additional work, the gradient boosting method in particular allows for very rapid training, tuning and inference due to how computationally efficient the XGBoost algorithm is.

Beyond new asset classes and time horizons, there are numerous possible directions for future research. One idea is to include additional predictive variables in a gradient boosting model, as the historical return series contains only so much information. Besides using VIX, promising variables may include trading volume data, with the idea that higher volume moves may give more of a hint about future volatility. A second option is to include other significant series as predictors, under the theory that what happens in one asset class can influence another. For instance, if predicting S&P 500 volatility, one may include historical returns (or volatilities) of US interest rates, commodities, or non-US equity indices. Increasing the sampling frequency of historical returns is yet another possible improvement. While we have used daily returns as features, many methods of volatility prediction utilize intraday values, which may give a fuller picture of the recent past; this could be combined with volume data or order book metrics for further explanatory power. A fourth extension could be to vary the amount of data used as the explanatory variables. We chose 126 days, but this was not a scientific selection, and it is unlikely to be ideal. Whatever the exact form of subsequent research, we hope we have shown the utility of tree-based methods in volatility prediction.

Author Contributions: All, M.L.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are openly available at https://github.com/MarinLolic/Volatility-Forecasting

Conflicts of Interest: The author declares no conflicts of interest.

References

Andersen, Torben G., Tim Bollerslev, Francis X. Diebold, and Paul Labys. 2003. Modeling and forecasting realized volatility. *Econometrica* 71: 579–625.

Ang, Andrew. 2014. *Asset Management: A Systematic Approach to Factor Investing.* New York: Oxford University Press, pp. 37-40.

Black, Fischer, and Myron Scholes. 1973. The pricing of options and corporate liabilities. *The Journal of Political Economy* 81: 637-654.

Bollerslev, Tim. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics* 31: 307-327.

Breiman, Leo, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees.* Boca Raton: Taylor & Francis Group.

Breiman, Leo. 1996. Bagging predictors. *Machine Learning* 24: 123-140.

Breiman, Leo. 2001. Random forests. *Machine Learning* 45: 5-32.

Bucci, Andrea. 2020. Realized volatility forecasting with neural networks. *Journal of Financial Econometrics 18*: 502-531.

Cornuejols, Gerard, Javier Pena, and Reha Tutuncu. 2018. *Optimization Methods in Finance*, Second Edition. Cambridge: Cambridge University Press, pp. 90-95.

Corsi, Fulvio. 2009. A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics* 7: 174-196.

Engle, Robert F. 1982. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica* 50: 987-1007.

Friedman, Jerome H. 2002. Stochastic gradient boosting. *Computational Statistics & Data Analysis* 38: 367-378.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*, Second Edition. New York: Springer, pp. 305-316.

Ilmanen, Antti. 2011. *Expected Returns.* Chichester: John Wiley & Sons Ltd, pp. 307-319.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2021. *An Introduction to Statistical Learning*, Second Edition. New York: Springer, pp. 15-42.

Lim, Ching Mun, and Siok Kun Sek. 2013. Comparing the performances of GARCH-type models in capturing the stock market volatility in Malaysia. *Procedia Economics and Finance* 5: 478-487.

Lopez de Prado, Marcos. 2018. *Advances in Financial Machine Learning.* Hoboken: John Wiley & Sons, pp. 103-110.

Mandelbrot, Benoit. 1963. The variation of certain speculative prices. *The Journal of Business* 36: 394-419.

Miller, Michael B. 2019. *Quantitative Financial Risk Management.* Hoboken: John Wiley & Sons, pp. 29-36.

Xiong, Ruoxuan, Eric P. Nichols, and Yuan Shen. 2015. Deep learning stock volatility with Google domestic trends. Working Paper. Available online: https://arxiv.org/abs/1512.04916 (Accessed on 25 September 2024).