Journal of Economics and Sustainable Development
Volume: 06 / N°: 01 (2023), P:680-701
EISSN:2773-2606 ISSN : 2661-7986

Machine learning for market volatility prediction

ATALLAH Amor [1*]
ECHAHID HAMMA LAKHDAR UNIVERSITY Of El OUED (Algeria),
atallah-amor@univ-eloued.dz

Received:19/12/2022 ***Accepted:02/02/2023*** ***Published:01/03/2023***

Abstract:

Market volatility prediction is one of the most commonly used terms in the trading market today. Price movements, market volatility, and trading risks are all represented by realized volatility. A small change in volatility affects the expected return on all assets.

In this research, To predict volatility, we will use the dataset provided by the Kaggle platform. Optiver is a leading global electronic market maker and is committed to continuously improving financial markets. improving access and prices for options. Exchange traded funds(ETFs), On numerous exchanges around the world, cash equities, bonds, and foreign currencies are traded. The prediction models we used in our study are LightGBM and XGBoost and CatBoost and Linear Regression, and we concluded some related works on forecasting volatility. Then we ran our models. The results show that the LightGBM model is the best among these models, as it achieved the lowest root mean square error percentage (RMSPE) score of: 0.286, And the highest score in the coefficient of determination R^2 is: 0.817, and the RMSPE for other models: XGBoost, CatBoost and Linear regression is, respectively: 0.303, 0.302 and 0.347, and the score of r^2 is also: 0.791, 0.784, 0.766.

Keywords: Machine learning, prediction market volatility.
JEL Classification: … ; … ; …

[*] ***Corresponding author.***

Introduction

Market Volatility is a very central and important topic in the financial literature and is of paramount importance in its many applications. It is an essential component of many investment decisions and is often the starting point for optimal portfolio allocations.

In 1952, Harry Markowitz laid the foundation for modern portfolio theory in which investors avoid risk and have an interest function that increases with expected return and decreases with volatility. Volatility is also a major input to the pricing of many derivatives. In fact, it is necessary to know the volatility of the underlying asset until the expiration date, to assess the fair value of the options. However, in recent years even derivatives with the same volatility have been presented as a basis, in these cases the definition and measurement of volatility must be specified in the derivative contracts themselves.

In risk management, volatility is associated with calculating the value at risk (VaR), the measurement of which at financial institutions has become a standard practice. Since the entry into force of the first Basel Convention in 1996, banks and trading venues are required to set aside a reserve capital of at least three times the capital at risk. Volatility also has broad consequences for the whole economy. Therefore, policy makers consider volatility as an indicator of uncertainty in the financial market. Besides, it negatively affects market liquidity, as when volatility increases, market liquidity usually decreases. Volatility is also crucial to hedging strategies. During tense market conditions, volatility increases, and this also leads to correlations between different securities. In these circumstances, derivative instruments can act as insurance against a sudden market downturn.

The importance of predicting volatility comes as a natural result of all of these previously mentioned reasons, and the literature on this topic has been extensive and many predictive models have been proposed in the past years.

We have seen a significant increase in the applications of machine learning techniques in many different sectors in recent years. We therefore presented this article which aims to predict market volatility using machine learning.

1-Defining and measuring volatility

Volatility is a statistical measure of the dispersion of returns for a specific security or index. The higher the volatility, the riskier the security because the price is less predictable**.**

Volatility is defined as the standard deviation or variance of returns from the same security or market index. There are several methods for determining or calculating the volatility of a stock or index.

Many aspects of volatility have been studied, including exchange rate volatility, oil price volatility, and community price volatility. Volatility is frequently measured by standard deviation, which refers to the daily, weekly, or monthly change in the value of a financial asset.

Volatility is a measure of the unexpected variability of asset returns over time. A security's or a market index's volatility is a measure of the spread of its returns across their mean. Typically denoted by σ, it is defined as the standard deviation of logarithmic returns observed over fixed time intervals**.**

The spread of all possible outcomes of an uncertain variable is referred to as volatility. In financial markets, we are frequently concerned with the spread of asset return. Statistically, volatility is often measured as sample standard deviation.

$$\boldsymbol{\sigma} = \sqrt{\frac{\mathbf{1}}{\boldsymbol{T-1}}\sum_{\boldsymbol{t=1}}^{\boldsymbol{T}}(\boldsymbol{r_t-\mu})^{\mathbf{2}}}$$

Where:

- $r_t = \log\left(\frac{p_t}{p_t-1}\right) \cong \frac{(p_t-p_{t-1})}{p_t}$ is the return on day t.
- p_t is the price at time t.
- μ μ is the mean of r_t.
- T is the period of time.

Variance σ^2 is sometimes used as a volatility metric. Because variance is simply the square of standard deviation, it doesn't matter which measure we use to compare the volatility of two assets; however, variance is less stable and desirable as an object for computer estimation and volatility forecast evaluation than standard deviation. Furthermore, standard deviation has the same unit of measure as the mean; if the mean is in dollars, the standard deviation is also expressed in dollars, whereas variance will be expressed in dollar square. For this reason, standard deviation is more convenient and intuitive when we think about volatility.

Volatility is expressed in annual terms. Since most of the variation in securities returns come from trading activity, practitioners consider a year as formed by 252 days, We annualize the standard deviation by multiplying it by 252 [6], which is typically the amount of days stocks are traded within a year. Assuming independent returns and constant volatility, After that, the annual volatility can be calculated as

$$\sigma_{annual} = \frac{\sigma_T}{\sqrt{\tau}}$$

Where τ is the length of time interval in years. For example, if we had to express daily volatility per annum:

$$\sigma_{annual} = \sigma_{daily}\sqrt{252}$$

The volatility mentioned above is commonly referred to as historical volatility. There are two kinds of volatility: realized volatility and implied volatility.

2-Realized volatility

With the availability of high-frequency data, suggest an alternative measure of volatility using all available high-frequency intraday data. This measure is so-called realized volatility. Comparable to the classical historical volatility, the realized volatility is a nonparametric expost estimate of return realisations over a fixed time interval. Particularly, this measure was built on the theory of continuous-time and arbitrage-free processes with the theory of quadratic variation. Let us expand the assumptions for the classical historical volatility estimator to include the following**.**

To lay out the fundamental concept and intuition, assume that the logarithmic $N \times 1$ vector price process, $logp_t$ follows a multivariate continuous-time stochastic volatility diffusion:

$$dlogp_t = \mu_{(t)}d_t + \sigma_t dw_t$$

where, w_t denotes a standard N-dimensional Brownian motion, the process for the N ×N positive definite diffusion matrix, μ_t is the drift term (continuous function) and σ_t is the instantaneous volatility that exaggerates the price change relative to dw_t .

In this time t 1 to t. The total return over the time period from $(t - \Delta t\ to\ t\ 0 \leq \Delta t \leq t)$, is written:

$$r_{(t,\Delta t)} = p_t - p_{(t,\Delta t)} = \int_{t-\Delta t}^{t} \mu_\tau d_\tau + \int_{t-\Delta t}^{t} \sigma_\tau\, w_t.$$

and the quadratic variation $QV(t - \Delta t)$.

$$QV(t - \Delta t) = \int_{t-\Delta t}^{t} \sigma_\tau^2 d_\tau.$$

Equations clearly shows that innovations to the mean component μ_t The return's sample path variation is unaffected. Intuitively, this is due the mean term, $\mu_t d_t$, In terms of second order properties, is of lower order than the diffusive innovations., $\sigma_t dw_t$. As a result, the cumulated returns from many high-frequency values over a short time interval can be neglected. Therefore, the variance in equation coincides with the quadratic variation $QV(t - \Delta t)$. Now, considering a discrete partition $(t - \Delta t + \frac{j}{n}, j = 1, \ldots., n.\Delta)$ of that the time interval $(t - \Delta t, t)$ is observed is observed equally at $\frac{j}{n}$,so, the realized volatility (the realized variance),RV, is defined like:

$$RV(t, \Delta t; n) = \sum_{j=1}^{n.\Delta t} r\left(t - \Delta t + \frac{j}{n}, \frac{1}{n}\right)^2.$$

Under semi-martingale theory, As the sampling frequency n increases, the realized variance measure converges in probability to the quadratic variation, i.e.

$$RV(t - \Delta t, t) = \sqrt{\sum_{j=0}^{N-1} r_{t-j\delta}}.$$

where,$(t - \Delta t, t)$ is the length of one day, $\delta = \frac{1}{N}, r_{t-j\delta} = p_{(t-j\delta)} - p_{(j-(j+1)\delta)}$, and returns are assumed to be sampled evenly at δ time step with N observations in that time period. Since we assume that the data is collected at equal-space, can be assembled at second, minute and hour frequencies.

2-Implied volatility

Implied volatility (IV), We find this expression definition in "investopedia". The expression implied volatility indicates to a metric that captures the market's perception of the likelihood of changes in a given security's price. Investors can use implied volatility to plan future moves and supply and demand, and often appoint it to price options contracts.

Implied volatility (IV) in the market indicates to the predicted magnitude, or one standard deviation (SD) scope, of potential movement away from the underlying price in a year's time.

Implied volatility $(\sigma_{civ,t}\ and\ \sigma_{piv,t})$ is a stock's current volatility as reflected by its option price. Because option pricing models cannot be inverted very easily, so implied volatility is calculated numerically. Implied volatility is estimated using the BS(BlackScholes) selection pricing model employing the bisection method as Displayed as follows:

$$\text{Volatility Estimate} = \boldsymbol{\sigma_t} + \frac{c - c_H}{c_L - c_H}(\boldsymbol{c_H} - \boldsymbol{c_L}).$$

Where c_L and c_H are the low and high volatility values respectively, c_L and c_H are the corresponding options values and C is the market price of the option.

An option pricing model can be used as a tool, to obtain the features of option prices, such as implied volatility, by relating the option price with the price of the underlying asset under the arbitrage free assumption to keep a fair market. The main thrusts behind generating the Black-Scholes model is to find a way to transform the market prices into an expression in terms of implied volatility.

Implied volatility is known to be a forecasting made by investors of the future movement of the underlying asset, i.e. it is their belief of how variable the stock is going to be between today and the maturity date. Since

the accurate forecast is crucial, reliable methods for approximating it are required. Given the observed option value V_{mkt}, implied volatility can be from:

$$V_{mkt} = BS(\sigma^*, S, K, T, r).$$

Since the Black-Scholes equation is monotonic with respect to σ ∗ , implied volatility exists and can be written implicitly by means of the inverse Black-Scholes function as

$$\sigma^* = BS^{-1}(V_{mkt}, S, K, T, r).$$

which does not have an analytic solution and thus, needs to be solved numerically.

3- Definition of machine learning:

Machine Learning (ML): It is a branch of Artificial Intelligence. Where machine learning algorithms build models based on data samples, called training data, to use them in order to make predictions or make decisions without directly programming them. It is the study of computer algorithms that can improve automatically through experience and with data.

Machine learning algorithms are used in several applications in different fields, such as medicine, voice and image recognition, email follow-up... These tasks are difficult to develop traditional algorithms to perform, or even useless for them. One of the most important areas of machine learning is making predictions using a computer that is related to computational statistics, in addition to data mining and exploratory data analysis through unsupervised learning. Some machine learning applications use data and neural networks in a way that mimics the functioning of the human brain.

4- Machine learning approaches:

Broadly, machine learning is split into four parts supervised learning, unsupervised learning, and semi-supervised learning, Reinforcement learning as shown in the following **f**igure:

Figure 1: Machine learning approaches

Source: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSRqDaKqKEtrb0o5glLMPv-FyIcDawfaZNpdbjuDpEkxA&s

4.1 Supervised learning:

In order to learn a general rule that defines inputs for outputs, the teacher gives the computer examples of the desired inputs and outputs. The support vector machine is a supervised learning model that splits data with a linear boundary into regions separated from each other, separating white circles from black circles. Supervised learning algorithms build a mathematical model of a set of data that contains the desired inputs and outputs. This data, called training data, is made up of a set of training examples. Each training example contains one or more desired inputs and outputs, also known as a supervisory signal. In the mathematical model, each training example is represented by a vector or matrix, called the feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms based on iterative optimization of an objective function learn a function that you can use to predict multiple outputs associated with new inputs. The optimization functionality of the algorithm allows the identification of new input outputs. We say that the algorithm has learned to perform the task entrusted to it if it works on improving the accuracy of its outputs or predictions over time. Supervised learning is further divided into two types based on the kind of problem Regression and Classification.

4.2 Unsupervised learning:

The machine learning algorithm is not given any labels. Unsupervised learning in itself can be a goal or an end (discovering hidden patterns in the data) and it can also be a means to an end (distinguished learning). Unsupervised learning algorithms take data that contains only the input, and find themselves a structure in this data, as aggregation or aggregation of data points. Therefore, these algorithms learn from test data that has not been labeled or ranked. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data to react based on whether or not they are present (commonality) in each new piece of data. The field of density estimation in statistics is one of the central applications of unsupervised learning. Although unsupervised learning includes other areas that include summarizing and explaining features of data.

4.3 Semi-Supervised Learning:

Semi-supervised learning lies between unsupervised learning (without any labeled training data) and supervised learning (with fully labeled training data). A computer program interacts with a dynamic environment in which it performs a specific goal as it navigates a problem space, and that program is provided with feedback similar to rewards, which it is trying to

magnify. Some training examples lack training labels, though machine learning researchers have found that unlabeled data can lead to a significant improvement in learning accuracy, if used in conjunction with a small amount of labeled data.

4.4 Reinforcement learning:

Reinforcement learning is an area concerned with how software agents take actions in an environment to maximize the idea of cumulative reward. Due to its generality, this field is studied in several different disciplines, including: information theory, operations research, genetic algorithms, statistics, and swarm intelligence. In machine learning, the environment is usually represented as a Markov decision process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. They do not assume knowledge of the exact mathematical model of MDP, and these algorithms are used when the exact models are not applicable. One of the areas of use of these algorithms in self-driving vehicles, and in learning to play against humans.

5- Machine Learning Models:

There are many machine learning models that are used, we will introduce some machine learning models:

5.1 XGBoost:

XGBoost is a scalable machine learning system. This model was developed by Chen and Guestrin (2015) who describe it in their paper as a scalable machine learning system based on the concept of gradient tree reinforcement. It gives advanced results in a wide range of problems, especially classification problems. Its scalability in all scenarios is the most important factor behind its success. The system runs 10 times faster than common solutions found on a single machine and can accommodate billions of examples in distributed or memory-limited settings. It has also recently gained in popularity as the best algorithm for many teams that have won machine learning competitions.

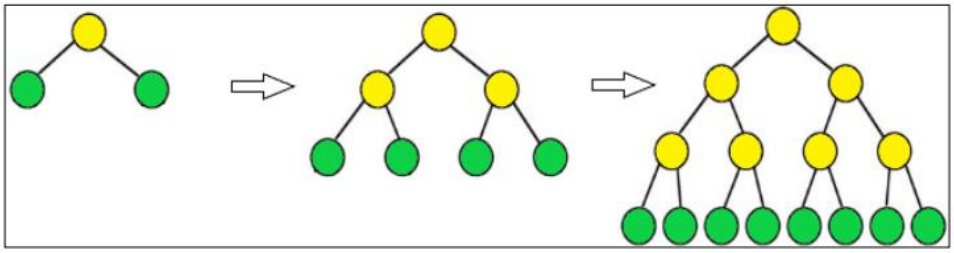
Chen and Guestrin listed some of the new innovative features that contribute to XGBoost's scalability, including:

- A new tree learning algorithm to deal with sparse data sets.
- A theoretically justified weighted graph procedure, allowing handling of instance weights in approximate tree learning.
- Parallel and distributed computing, to make the training period much shorter, to enable us to make faster model explorations.

- Exploiting out-of-core computations allows data scientists to process large amounts of data.

Through the growth of level-wise trees the XGBoot tree expands. The corresponding figure shows an example of the way XGBoot tree expands.

Figure 2: XGBoost tree expansion method



Source: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSRqDaKqKEtrb0o5glLMPv-FyIcDawfaZNpdbjuDpEkxA&s

Yellow circles denote an older level of leaves, which have already been calculated and green circles denote the level of new leaves that are added to the tree.

5.2 LightGBM:

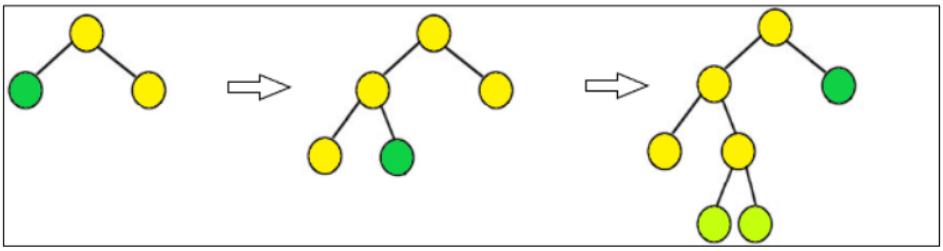
LightGBM is an acronym for Light Gradient Boosting Machine developed by Ke et al. (2017), another GBDT model, can significantly outperform XGBoost in terms of computing speed and memory consumption. This algorithm contains two new technologies: gradient-based one-sided sampling and exclusive feature aggregation to handle a large number of instances Data and a large number of features respectively. In scenarios involving large amounts of data and a number of features. The LightGBM model uses XGBoost as a baseline, while approaching the classification problem in a completely different way, through the use of the two new technolog**.**

In scenarios involving large amounts of data and a number of features. The LightGBM model uses XGBoost as a baseline, while approaching the classification problem in a completely different way, through the use of the two new technologies.

Gradient-based one-sided sampling necessitates that the model omits most examples where we expect the gradient weight to be smaller. This may help us avoid going down the branches, which may be less important. Data

states with different gradient values have a different effect on the expected information acquisition. This is how trees are expanded in LightGBM, sheet by sheet. The following figure illustrates this:

Figure 3: LightGBM method



Source: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSRqDaKqKEtrb0o5glLMPv-FyIcDawfaZNpdbjuDpEkxA&s

The yellow circles denote older leaves, which have already been explored, the green circles denote the current leaf that is being considered.

5.3 CatBoost:

CatBoost is a model proposed by Prokhorenkova et al. (2019) Focusing more on categorical features, they argued that the CatBoost algorithm introduces two new core functions, a rotation-driven rank-boosting algorithm and another algorithmic approach specifically for dealing with categorical features. They show that CatBoost combats the problem of exponential feature set growth by using a greedy method At each new split of the current tree. It is also capable of handling scenarios where the number of classes is too large for other GBDT models. CatBoost addresses this issue with these three steps:

- It divides the data into random subgroups as a first stage.
- Converts labels to integers.
- Finally, it converts the remaining categorical attributes to numerical values.

The resulting data set should then be ready for use by the CatBoost model. The exact technical specifications of each step largely depend on what the developers specify. CatBoost is best when we are dealing with non-numeric data as it can convert categorical values to integers in an optimal way. Performance may vary greatly depending on the types of features implemented because CatBoost was developed to handle categorical features. It's a good idea to implement two versions of the CatBoost model to quantify the difference in performance, with one being trained on a

categorical feature dataset and the other on a hot-coded single feature dataset

.

5.4 Linear Regression:

Linear regression is a model that has been around for a long time and is one of the most important tools in the field of statistics. This linear regression model can be represented by the following mathematical expression:

$$X = (X_1, X_1, \dots, X_p).$$
$$\beta = (\beta_1, \beta_2, \dots, \beta_p).$$
$$\hat{Y} = \hat{\beta}_0 + \sum_{i=1}^{p} X_i \beta_i.$$

$X_i$represents the model input variables, and Y represents the model output variable. β_i, $i = 1, 2, \dots, p$, are the model parameters to be estimated.The term β0 is the intersection, or the so-called bias in machine learning. It is often more convenient to include the constant variable 1 in X , include β_0 in the vector of coefficients β , and then write the linear model in vector form as an internal product:

$$\hat{\mathrm{Y}} = \mathrm{X}^{\mathrm{T}} \hat{\beta}.$$

Estimation using the least squares method. In this approach the most common method for estimating the model parameters is, and an estimate is made in order to reduce the remaining sum of squares:

$$\mathrm{RSS}(\beta) = \sum_{\mathrm{i=1}}^{\mathrm{N}} (y_i - x^T \hat{\beta})^2.$$

writing the formula in matrix notation we have:

$$\mathrm{RSS}(\beta) = (Y - X\beta)^2 (Y - X\beta).$$

where X is an $N * P$ matrix with each row and input vector, and y is an N vector of the outputs in training set. differentiating in order of β and equal to zero we get to the equations:

$$\mathrm{X}^{\mathrm{T}}(\mathrm{Y} - \mathrm{X}\beta) = 0.$$

$$\beta = (\mathrm{X}^{\mathrm{T}} \mathrm{X})^{-1} \mathrm{X}^{\mathrm{T}}.$$

6- Experimental results

6.1 Dataset :

The regression applies to several types of data from several fields (finance, biology,chemistry, marketing etc.)

We started collecting the data by using the information provided on the kaggle.com site. Our study contains stock market data relevant to the practical execution of trades in the financial markets. In particular, it includes order book snapshots and executed trades. With one second resolution, it provides a uniquely fine grained look at the microstructure of modern financial markets.

6.1.1 Book parquet:

A parquet file partitioned by stock_id. Provides order book data on the most competitive buy and sell orders entered into the market. The top two levels of the book are shared. The first level of the book will be more competitive in price terms, it will then receive execution priority over the second level.

- stock_id - ID code for the stock. Not all stock IDs exist in every time bucket. Parquet coerces this column to the categorical data type when loaded; you may wish to convert it to int8.
- time_id - ID code for the time bucket. Time IDs are not necessarily sequential but are consistent across all stocks.
- seconds_in_bucket - Number of seconds from the start of the bucket, always starting from 0.
- bid_price[1/2] - Normalized prices of the most/second most competitive buy level.
- ask_price[1/2] - Normalized prices of the most/second most competitive sell level.
- bid_size[1/2] - The number of shares on the most/second most competitive buy level.
- ask_size[1/2] - The number of shares on the most/second most competitive sell level.

Figure 4: DataFrame of Book_[train/test].parquet

	time_id	seconds_in_bucket	bid_price1	ask_price1	bid_price2	ask_price2	bid_size1	ask_size1	bid_size2	ask_size2	stock_id
0	5	0	1.001422	1.002301	1.00137	1.002353	3	226	2	100	0
1	5	1	1.001422	1.002301	1.00137	1.002353	3	100	2	100	0
2	5	5	1.001422	1.002301	1.00137	1.002405	3	100	2	100	0
3	5	6	1.001422	1.002301	1.00137	1.002405	3	126	2	100	0
4	5	7	1.001422	1.002301	1.00137	1.002405	3	126	2	100	0
5	5	11	1.001422	1.002301	1.00137	1.002405	3	100	2	100	0
6	5	12	1.001422	1.002301	1.00137	1.002405	3	126	2	100	0
7	5	14	1.001422	1.002301	1.00137	1.002405	3	126	2	100	0
8	5	15	1.001422	1.002301	1.00137	1.002405	3	126	2	100	0
9	5	16	1.001422	1.002301	1.00137	1.002405	3	126	2	100	0

Source: Extracted from the python

6.1.2 Trade parquet:

A parquet file partitioned by stock_id. Contains data on trades that actually executed. Usually, in the market, there are more passive buy/sell intention updates (book updates) than actual trades, therefore one may expect this file to be more sparse than the order book.

- stock_id - Same as above.
- time_id - Same as above.
- seconds_in_bucket - Same as above. Note that since trade and book data are taken from the same time window and trade data is more sparse in general, this field is not necessarily starting from 0.
- price - The average price of executed transactions happening in one second. Prices have been normalized and the average has been weighted by the number of shares traded in each transaction.
- size - The sum number of shares traded.
- order_count - The number of unique trade orders taking place.

Figure 5: DataFrame of Trade_[train/test].parque

	time_id	seconds_in_bucket	price	size	order_count	stock_id
0	5	21	1.002301	326	12	0
1	5	46	1.002778	128	4	0
2	5	50	1.002818	55	1	0
3	5	57	1.003155	121	5	0
4	5	68	1.003646	4	1	0
5	5	78	1.003762	134	5	0
6	5	122	1.004207	102	3	0
7	5	127	1.004577	1	1	0
8	5	144	1.004370	6	1	0
9	5	147	1.003964	233	4	0

Source: Extracted from the python

6.1.3 Train:

Provides the mapping between the other data files and the submission file.

- stock_id - Same as above, but since this is a csv the column will load as an integer instead of categorical.
- time_id - Same as above.
- target - The realized volatility computed over the 10 minute window following the feature data under the same stock/time_id. There is no overlap between feature and target data.

Figure 6: DataFrame of Train.csv

	stock_id	time_id	target
0	0	5	0.004136
1	0	11	0.001445
2	0	16	0.002168
3	0	31	0.002195
4	0	62	0.001747
5	0	72	0.004912
6	0	97	0.009388
7	0	103	0.004120
8	0	109	0.002182
9	0	123	0.002669

Source: Extracted from the python

There are 112 types of stock_id, 3830 types of time_id, and 414287 types of target.

6.2 Data processing :

The dataset includes the Weighted averaged price and log returns and Realized volatility.

6.2.1 Weighted averaged price:

The order book is also one of the primary source for stock valuation. A fair bookbased valuation must take two factors into account: the level and the size of orders. In this competition we used weighted averaged price, or WAP, to calculate the instantaneous stock valuation and calculate realized volatility as our target.

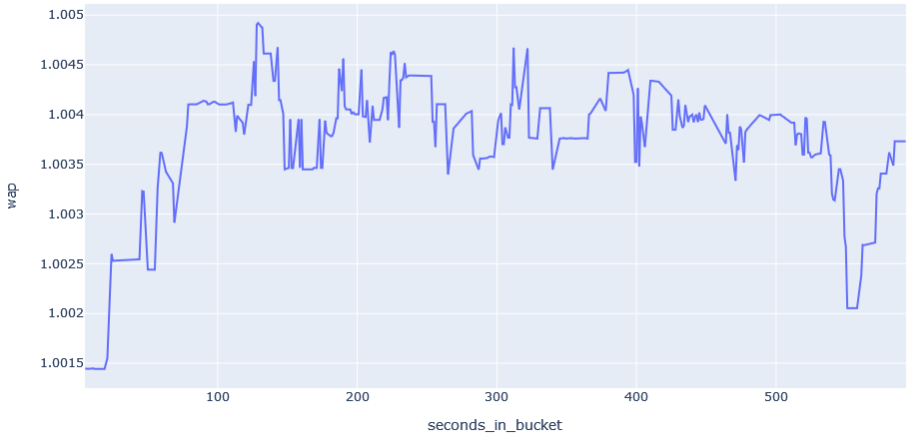
To calculate realized volatility first we calculate the weighted average price using the formula of WAP can be written as below, which takes the top level price and volume information into account:

$$\mathrm{W\,AP} = \frac{(\mathrm{BibPrice1} * \mathrm{AskSize1} * \mathrm{AskPrice1} * \mathrm{BidSize1})}{(\mathrm{BidSize1} * \mathrm{AskSize1})}$$

As you can see, if two books have both bid and ask offers on the same price level respectively, the one with more offers in place will generate a lower stock valuation, as there are more intended seller in the book, and more seller implies a fact of more supply on the market resulting in a lower stock valuation.

Note that in most of cases, during the continuous trading hours, an order book should not have the scenario when bid order is higher than the offer, or ask, order. In another word, most likely, the bid and ask should never be in cross.

Figure 7: WAP stock_id_0,time_id_5



Source: Extracted from the python

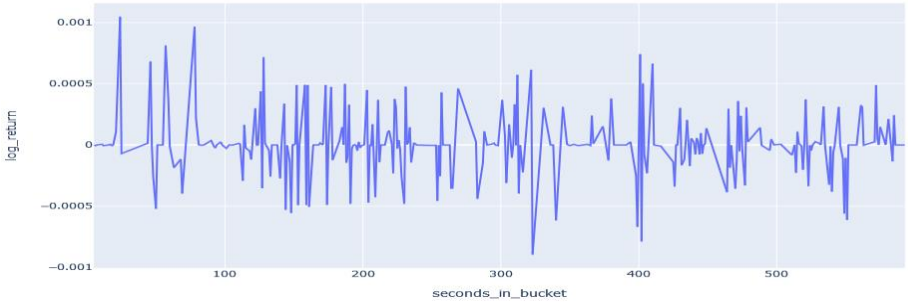
6.2.2 Calculate Log returns:

Returns are widely used in finance, however log returns are preferred whenever some mathematical modelling is required. Calling P_t the price of the stock S at time t, we can define the log return between t_1 and t_2 as:

$$r_{t1,t2} = \log\left(\frac{P_{t1}}{P_{t2}}\right)$$

Usually, we look at log returns over fixed time intervals, so with 10-minute log return we mean $r_t = r_{t-10\,min,t}$. Where we use WAP as price of the stock to compute log returns.

Figure 7: Log return of stock_id_0,time_id_5



Source: Extracted from the python

6.2.3 Calculate Realized volatility:

When we trade options, a valuable input to our models is the standard deviation of the stock log returns. The standard deviation will be different for log returns computed over longer or shorter intervals, for this reason it is usually normalized to a 1-year period and the annualized standard deviation is called volatility Which we talked about it in the first chapter.

We will compute the log returns over all consecutive book updates and we define the realized volatility, σ, as the squared root of the sum of squared log returns.

$$\sigma = \sqrt{\sum_{t} r_{t-1,t}^2}$$

We want to keep definitions as simple and clear as possible, So we are not annualizing the volatility and we are assuming that log returns have 0 mean.

6.3 Methodology:

In order to conduct a fair comparison between the four different models machine learning procedures to predict to predict the market volatility of Optiver. LightGBM(Light Gradient Boosting Machine) and XGBoost and CatBoostRegressor and Linear Regression methods are used. The performance of the models is measured by two metrics:

RMSPE: Root Mean Square Error, it is a measure of the deviation between the observed value and the real value and it is often used as a standard for the prediction results of machine learning models.

$$\text{RMSP E(X,)} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(X_i - Y_i)^2}$$

Where (X_i) is the prediction value at the time i, (Y_i) is the and ground truth value at time t and m is the total number of the train or test data.

$\boldsymbol{R^2}$ **score:** is a metric that tells the performance of your model, not the loss in an absolute sense that how many wells did your model perform.

$$R^2 \text{ score} = 1 - \frac{SSr}{SSm}$$

where SSr is squared sum error of regression line and SSm is squared sum error of mean line.

The machine learning library sklearn for Python is used to create the addressed machine learning models.

For all four learning methods(LightGBM and XGBoost and CatBoost and Linear Regression), multiple models are created, and two tests are carried out. In many cases, the dataset is first shuffled and then split into training and testing, and this is after calculating (WAP,log(return),realized volatility), we used the machine learning library sklearn for Python to split

dataset for the best results. Afterwards, fitting the model is done, and then the fitted model is used to predict the realized volatility. The predictions are then compared with the actual values, and the errors are calculated.

6.4 Features of model:

Features are a vital component for machine learning models. They are the foundation on what the predictions are made. In this case, We used three files (book-train, tradetrain, train.cvc) to prepare the features as shown in figure as follows:

Figure 8: Dataframe after preparing data

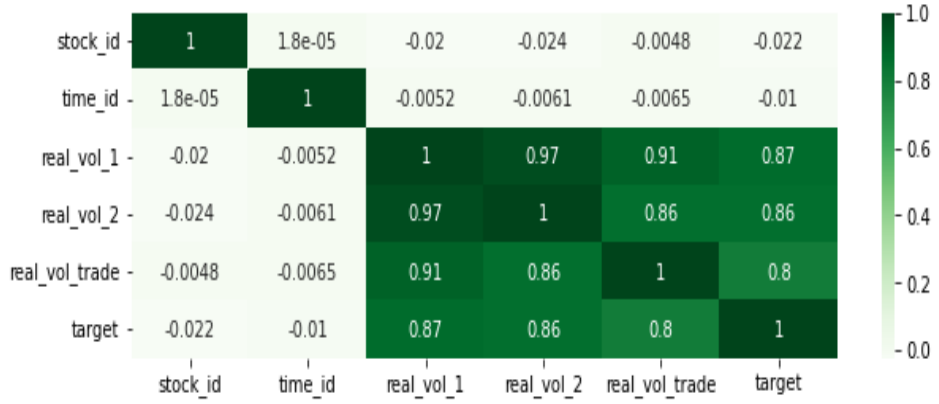
	stock_id	time_id	real_vol_1	real_vol_2	real_vol_trade	target
0	32	5	0.002370	0.002919	0.001805	0.002649
1	32	11	0.001249	0.001784	0.000805	0.000688
2	32	16	0.002012	0.002388	0.001762	0.002431
3	32	31	0.001421	0.001984	0.001236	0.001881
4	32	62	0.001308	0.001663	0.000990	0.001463

Source: Extracted from the python

In this dataframe, we have six features(variables) stock-id, time-id, target, and (realvol1, real-vol2, real-vol-trade) that we obtained from calculate function (WAP and log return).

The linear correlation matrix presented on the below of figure shows a linear correlation between , time-id, target, real-vol1, real-vol2, real-vol-trade as follows:

Figure 8: : Correlation between the 6 features



Source: Extracted from the python

6.5 Models Comparison:

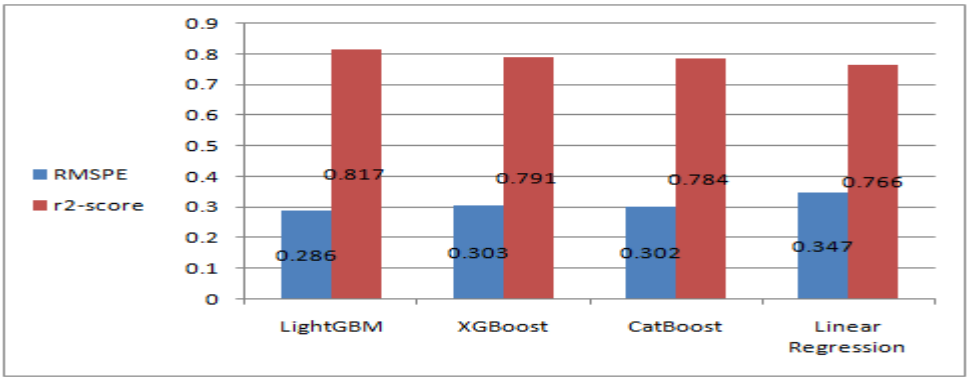
In this section, the four models can now be compared. The comparison of the models are displayed in Table 1 as follows

Table 1: Best performing Machine learning models comparison

model/metric	$r^2 - score$	RMSPE
LightGBM	0.817	0.286
XGBoost	0.791	0.303
CatBoost	0.784	0.302
Linear Regression	0.766	0.347

Source: Extracted from the python

Figure 9: Histogram Comparison between models



Source: Extracted from the excel

We note through the table 3.1, the best result of each metric is colored gray. As seen, each model excels in a specific metric. the result shows that model LightGBM has a lowest RMSPE score that is 0.286 and a highest $r^2 - score$ is 0.817, and the RMSPE of other models such as XGBoost, CatBoost, Linear Regression are respectively 0.303, 0.302, 0.347, and $r^2 - score$ are 0.791, 0.784, 0.766. the best model is LightGBM.

Conclusion

Market volatility prediction is one of the most commonly used terms in the trading market today. Price movements, market volatility, and trading risks are all represented by realized volatility. A small change in volatility affects the expected return on all assets.

The goal of this article was to predict market volatility with machine learning models, compare the results and determine which approach is more efficient. In particular, our analysis covered the LightGBM , XGBoost , CatBoost and Linear Regression.

A preliminary theoretical discussion allowed us to focus our empirical research on the the LightGBM, XGBoost , CatBoost and Linear Regression. These were compared based on their forecasting accuracy of the realized volatility.

The prediction models we used in our study are LightGBM , XGBoost , CatBoost and Linear Regression, We deduced some related works about the volatility forecasting. And then we ran our models.The result shows that model LightGBM has a lowest RMSPE score that is 0.286 and a highest r2-

score is 0.817. The RMSPE of other models such as XGBoost, CatBoost, Linear Regression are respectively 0.303, 0.302, 0.347, and r2-score are 0.791, 0.784, 0.766. The result of lightGBM was the best compared to other models.

Through this study, we find that LightGBM is the best prediction model to predict realized volatility.

Bibliography

[1] A. Nõu, D. Lapitskaya, M. H. Eratalay, and R. Sharma, "Predicting stock return and volatility with machine learning and econometric models: a comparative case study of the baltic stock market," Available at SSRN 3974770, 2021.

[2] A. Bucci et al., "Forecasting realized volatility: a review," Journal of Advanced Studies in Finance (JASF), vol. 8, no. 16, pp. 94–138, 2017.

[3] R. J. Shiller, Market volatility. MIT press, 1992.

[4] S.-H. Poon, A practical guide to forecasting financial market volatility. John Wiley & Sons, 2005.

[5] S. Muzzioli, "The relation between implied and realised volatility: are call options more informative than put options? evidence from the dax index options market," 2007.

[6] S. Romano, "Machine learning for volatility forecasting," 2021.

[7] P. A. C. Luong, Measuring and Modelling the Volatility of Financial Time Series. PhD thesis, Curtin University, 2016.

[8] T. G. Andersen, T. Bollerslev, F. X. Diebold, and P. Labys, "Modeling and forecasting realized volatility," Econometrica, vol. 71, no. 2, pp. 579–625, 2003. [9] T. G. Andersen, T. Bollerslev, F. X. Diebold, and H. Ebens, "The distribution of realized stock return volatility," Journal of financial economics, vol. 61, no. 1, pp. 43– 76, 2001.

[10] E. Rahimikia, S. Zohren, and S.-H. Poon, "Realised volatility forecasting: Machine learning via financial word embedding," Available at SSRN 3895272, 2021.

[11] T. G. Andersen and L. Benzoni, "Realized volatility, working paper 2008-14," 2008.

[12] S.-w. He, J.-g. Wang, and J.-a. Yan, Semimartingale theory and stochastic calculus. Routledge, 2019. [13] P. Padhi and I. Shaikh, "On the relationship of implied, realized and historical volatility: evidence from nse equity index options," Journal of Business Economics and Management, vol. 15, no. 5, pp. 915–934, 2014.

[14] P. Tankov, Financial modelling with jump processes. Chapman and Hall/CRC, 2003.

[15] J. D. MacBeth and L. J. Merville, "An empirical examination of the black-scholes call option pricing model," The journal of finance, vol. 34, no. 5, pp. 1173–1186, 1979.

[16] S. Arziyev, "Simulations of implied volatility and option pricing using neural networks and finite difference methods for heston model," 2020.

[17] S. Liu, C. W. Oosterlee, and S. M. Bohte, "Pricing options and computing implied volatilities using neural networks," Risks, vol. 7, no. 1, p. 16, 2019.

[18] T. M. Mitchell, "Artificial neural networks," Machine learning, vol. 45, pp. 81–127, 1997.

[19] T. M. Mitchell, "Does machine learning really work?," AI magazine, vol. 18, no. 3, pp. 11–11, 1997. [20] B. He, K.-i. Shu, and H. Zhang, "Machine learning and data mining in diabetes diagnosis and treatment," in IOP Conference Series: Materials Science and Engineering, vol. 490, p. 042049, IOP Publishing, 2019.