Realized Volatility Prediction

STATS 507 Final Project

Ching Hin Yeung
Department of Statistics
University of Michigan – Ann Arbor
Ann Arbor, Michigan, USA
Email: chyeung@umich.edu
Link: https://github.com/chyeungUM/STATS-507-Final-Project.git

***Abstract*—This project develops a short-term volatility forecasting model using high-frequency order-book and trade data from the Optiver Kaggle competition. A stacked architecture combining LightGBM (LGBM) and Support Vector Machines (SVM) is employed, supported by extensive feature engineering derived from weighted average prices, log returns, etc. Model performance is evaluated using 5-fold cross-validation and the RMSPE metric. Results show that the stacked model outperforms both base learners and naive benchmarks, benefiting from the complementary linear and nonlinear information captured by SVM and LGBM. The findings demonstrate that integrating heterogeneous models enhances predictive accuracy for high-frequency volatility forecasting.**

***Index Terms*—Volatility Forecasting, LGBM, SVM, Stacking, RMSPE, High-Frequency Finance, Optiver**

I. Introduction

A. Background

Volatility is a core concept in finance, reflecting the scale of price fluctuations and acting as a key indicator of market uncertainty. High-volatility periods often feature sharp price swings and elevated risk, while low-volatility regimes are more stable. Because option values depend heavily on the volatility, accurate volatility forecasts are crucial for pricing, hedging, and risk management.

Optiver, a leading global market maker, supports market quality by improving price discovery and liquidity across a wide range of instruments, from equities to bonds. To do so, the firm relies on advanced models to estimate volatility and ensure fair, competitive pricing. Yet achieving high precision in volatility modeling remains a persistent challenge. Through initiatives like Kaggle competitions, Optiver encourages innovation and pushes the boundaries of market-making research.

B. Motivation

My interest in this project is motivated by recent experience in trading, which has heightened my curiosity about the micro- and macro-structural mechanisms that govern markets. I seek to advance from a preliminary understanding to a more rigorous, data-driven framework for interpreting market behavior, with particular attention to the determinants of price dynamics and volatility. This Kaggle competition offers an appropriate setting to pursue these aims, enabling the implementation of quantitative modeling within a realistic market context.

C. Project Goal

The primary objective of this project is to develop a pretrained predictive model capable of estimating short-term volatility over the subsequent 10-minute interval. Such a model holds practical relevance for assessing short-horizon trading risk in real-world market. To enhance predictive accuracy and robustness, the analysis will incorporate additional feature-engineering procedures, including the construction of interaction terms. Following model development, variable-importance analysis will be conducted to quantify each feature's contribution to overall predictive performance. This examination not only offers insight into the determinants of short-term (10-minute) volatility but also informs feature-selection decisions aimed at reducing dimensionality and computational complexity.

D. Literature Review

1) Literatures: Volatility prediction has become increasingly important, as it serves as a key indicator of expected risk. Dierckx, Davis, and Schoutens (2020) show that machine learning methods—including Logistic Regression, Support Vector Machines, and AdaBoost—can surpass traditional econometric models in forecasting implied volatility. Many studies further highlight the effectiveness of deep learning approaches. For instance, Lin (2018) applied GARCH-type models to the SSE Composite Index, while Wang et al. (2019) integrated news-based sentiment into a hybrid neural network for stock volatility prediction. Beyond equities, Zhao (2020) used a LSTM model on cryptocurrency data, and Vidal and Kristjanpoller (2020) combined CNN and LSTM architectures to model gold price volatility. Together, these literatures demonstrate the strong capacity of deep learning techniques to capture the complex dynamics of financial time series.

2) Samples: Kaggle's leaderboard indicates that although many participants adopted variations of nyanp's approach, none surpassed his performance on the private test. This reflects the strength of his ensemble, which combines LGBM, MLP, and 1D-CNN, supported by reverse-engineered time-series validation and Nearest-Neighbor. Other top solutions—such as chumajin's TabNet–LGBM ensemble and the Neural Network–TabNet–LGBM mixture by shigeria and

Yuki Hakamada—highlight a consistent pattern: high accuracy arises from integrating complementary model architectures. LGBM appears as a common backbone due to its ability to handle nonlinearities and high-dimensional features, while pairing effectively with linear components. Motivated by these insights, this project adopts a LGBM–SVM stacking to assess potential gains from model complementarity.

II. Method

A. Data Understanding

Data provided by Kaggle consisted of six files: `train.csv`, `test.csv`, `book_train.parquet`, `book_test.parquet`, `trade_train.parquet`, and `trade_test.parquet`. The details of the columns in each file are as follows:

1) *[train/test].csv:*
   a) *stock_id:* Stock identifier.
   b) *time_id:* Time identifier (in seconds).
   c) *target:* Realized volatility over the next 10 minutes.
2) *book_[train/test].parquet:*
   a) *stock_id:* Stock identifier.
   b) *time_id:* Time identifier (in seconds).
   c) *seconds_in_bucket:* Seconds since a stock is available.
   d) *bid_price[1/2]:* Top two bid prices.
   e) *bid_size[1/2]:* Top two bid volumes.
   f) *ask_price[1/2]:* Top two ask prices.
   g) *ask_size[1/2]:* Top two ask volumes.
3) *trade_[train/test].parquet:*
   a) *stock_id:* Stock identifier.
   b) *time_id:* Time identifier (in seconds).
   c) *seconds_in_bucket:* Seconds since a stock is available.
   d) *price:* Volume-weighted transaction price.
   e) *size:* Total volume traded.
   f) *order_count:* Number of unique trade orders.

The goal of this project is to forecast 10-minute stock volatility across multiple sectors. The `train.csv` contains 428,932 unique (`stock_id`, `time_id`) pairs across 112 stocks, with most stocks having 3,830 time points. These identifiers link to the book and trade parquet files, which provide detailed information such as bid/ask prices and sizes, as well as executed trade quantities.

Given the weighted average price formula $S_{i,t}$ $[i = 1/2]$, the corresponding log returns $r_{i,t,t-1}$, and realized volatility σ_i can be computed sequentially as follows:

$$S_{i,t} = \frac{\textit{Bid Price}_{i,t} \times \textit{Ask Size}_{i,t} + \textit{Ask Price}_{i,t} \times \textit{Bid Size}_{i,t}}{\textit{Bid Size}_{i,t} + \textit{Ask Size}_{i,t}}$$

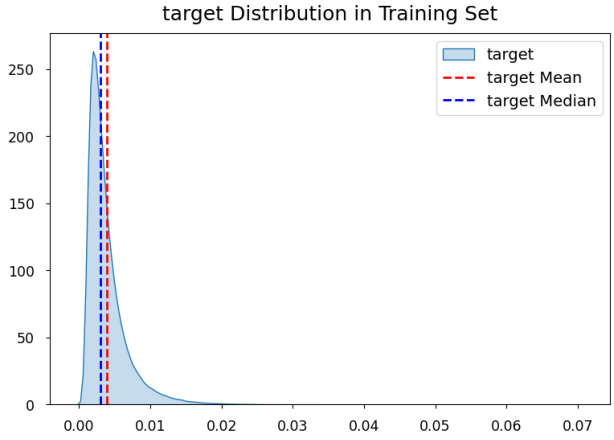
$$r_{i,t,t-1} = \log\left(\frac{S_{i,t}}{S_{i,t-1}}\right)$$

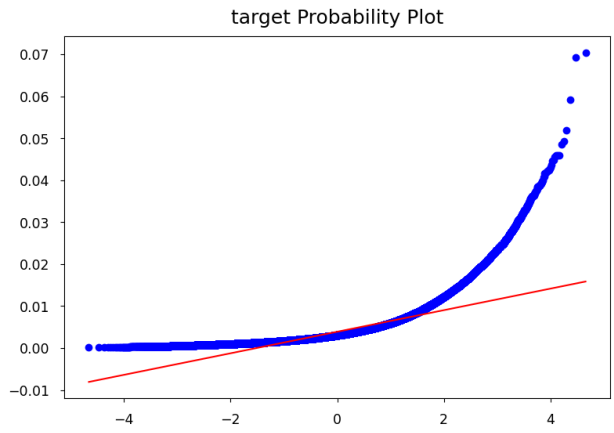
$$\sigma_i = \sqrt{\sum_t r^2_{i,t,t-1}}$$

To characterize the target volatility, we summarize its key distributional properties.

Mean	Std	Skew	Kurtosis	Missing
0.0039	0.0029	2.8226	14.9611	0/428,932

Min	25%	Median	75%	Max
0.0001	0.0020	0.0030	0.0047	0.0703





The statistics and plots show that the volatility distribution is asymmetric and heavy-tailed, a common feature of financial data. The close mean and median indicate that most observations fall in a low-volatility range, while the pronounced skewness (2.82) and kurtosis (14.96) reveal a long right tail driven by occasional extreme values. Density and QQ plots confirm clear departures from normality, especially in the upper quantiles. These characteristics suggest that effective modeling must accommodate non-Gaussian behavior, making flexible nonlinear methods more suitable than traditional approaches.

B. Methodology

1) Feature Engineering: Using the information from the trade and order book, we construct weighted average prices, log returns, and realized volatilities (1 and 2). To enhance predictive performance, we generate additional features derived from these quantities. Fundamental statistical descriptors—such as means, sums, standard deviations, balances,

imbalances, spreads, minima, and maxima—are appended to the dataset.

To capture short-horizon dynamics, these statistics are further aggregated over forward-looking windows of 150, 300, and 450 seconds, which correspond to the 25th, 50th, and 75th percentiles of the 10-minute forecast window. This procedure yields over 240 supplementary predictors. For example, the feature `log_return2_realized_volatility_150_mean_stock` represents the mean realized volatility computed from the second log returns over the next 150 seconds for each (`stock_id`, `time_id`) pair.

2) Method of Analysis: The study employs LGBM and SVM, combined within a stacking framework to leverage their complementary strengths. LGBM captures nonlinearity and complex feature interactions through gradient boosting, while SVM provides a robust linear margin-based component suitable for high-dimensional settings. A Meta–LGBM regressor is trained on the base-model predictions, enabling the stacked architecture to integrate nonlinear flexibility with linear stability to improve predictive accuracy.

Because Kaggle withholds all true testing data—including `trade_test` and `book_test`—model performance is assessed using 5-fold cross-validation and the Root Mean Squared Percentage Error (RMSPE). Since it penalizes proportional rather than absolute errors, better reflecting relative prediction accuracy across stocks with different volatility scales. The RMSPE is defined as:

$$\text{RMSPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}. \tag{1}$$

In addition, simple benchmark models provide useful reference points for performance evaluation. Predicting volatility using the overall mean yields an RMSPE of 1.11033, while predicting with the mean and median by stock produces RMSPE values of 0.789618 and 0.589135, respectively. These baselines serve as comparison standards for assessing the effectiveness of the base models as well as the stacked LGBM–SVM model.

III. Results

A. Models and Error

Using 5-fold cross-validation with `random_state=507`, the LGBM model is trained with the following key hyperparameters: `n_estimators=500`, `learning_rate=0.05`, `max_depth=-1`, `subsample=0.8`, `colsample_bytree=0.8`, and early stopping after 50 rounds without improvement. The SVM model uses a linear kernel with `C=0.1`, `epsilon=0.0`, and `random_state=507`. The predictions from these two base learners are then stacked using a Meta-LGBM regressor configured with: `n_estimators=300`, `learning_rate=0.05`, `max_depth=3`, `subsample=0.9`, `colsample_bytree=1.0`. The fold-level performance is summarized below:

	RMSPE			Time (sec)		
Fold	LGBM	SVM	Stacked	LGBM	SVM	Total
1	0.252438	0.526088	0.250237	17.4910	203.3797	221.5362
2	0.245996	0.461298	0.243640	15.8219	206.6216	223.2701
3	0.236605	0.480501	0.234835	16.6751	211.4381	228.9436
4	0.254969	0.654408	0.250061	16.9743	214.0968	231.9090
5	0.250669	0.479376	0.248347	16.2473	213.2475	230.3195

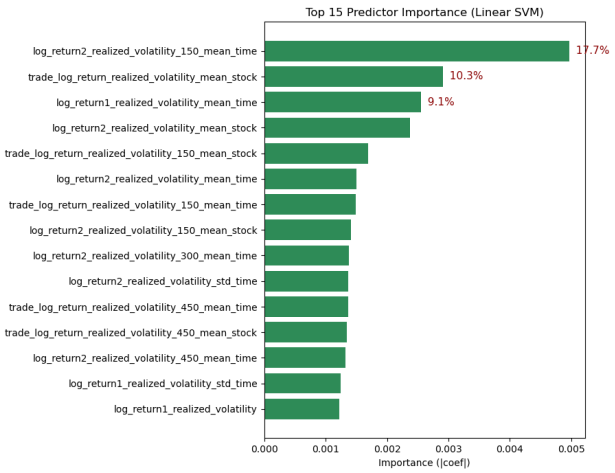
Both LGBM and SVM outperform the baseline RMSPEs obtained using stock-level mean and median predictions. Although LGBM consistently outperforms SVM, the stacked model achieves an additional reduction of approximately 0.002–0.003 in RMSPE, demonstrating the benefit of combining linear and nonlinear components within the stacking architecture.

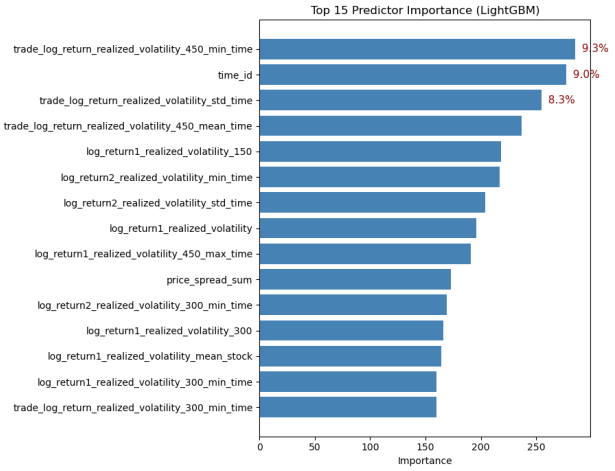
Regarding computational cost, SVM dominates the runtime, requiring roughly ten times the training time of LGBM. This is expected, as linear SVM scales on the order of O(n×p), which becomes substantial with 428,932 observations and over 240 predictors. Nevertheless, the improved predictive accuracy from stacking justifies the additional computation. The overall performance metrics are:

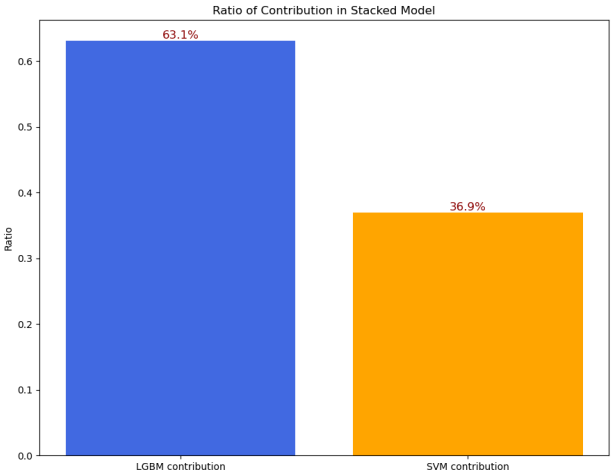
Metric (RMSPE)	LGBM	SVM	Stacked
Cross-Validation	0.248135	0.520334	0.245424
Out-of-Fold	0.248220	0.525071	0.245493

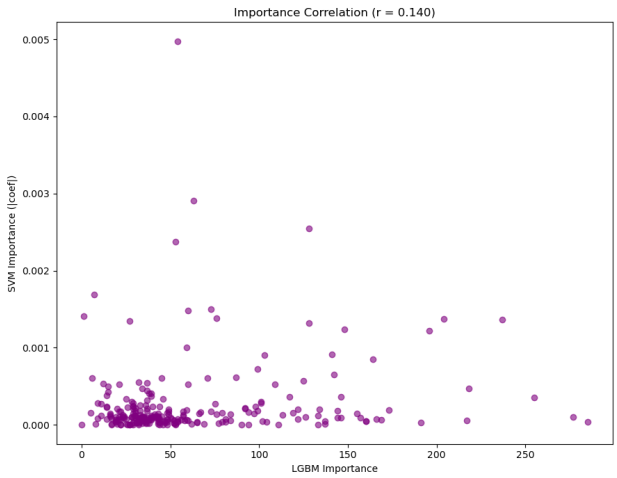
The stacked model consistently outperforms both base LGBM and SVM model, reinforcing confidence that the full model will deliver similar improvements when trained on all available data. Accordingly, the cross-validated parameters are used to fit the complete model, and the resulting preprocessing objects and trained estimators are saved as artifacts: `ordinal_encoder.pkl`, `imputer.pkl`, `scaler.pkl`, `lgbm_base.pkl`, `svm_base.pkl`, `meta_lgbm.pkl`, packaged in `trained_models.zip`.

B. Feature Importance









The feature-importance results indicate that LGBM and SVM draw on notably different predictors, consistent with their distinct modeling structures. The linear SVM concentrates its weight on realized-volatility features derived from log returns, especially those aggregated over short horizons such as 150 seconds. These features dominate the SVM's signal and illustrate its tendency to capture stable linear relationships in the data. In contrast, LGBM emphasizes a wider and more diverse set of predictors, including `time_id`, `trade_log_return_realized_volatility_450_min_time`, and extremum-based measures over 300–450-second windows. This distribution reflects LGBM's capacity to learn nonlinear patterns and interaction effects that extend beyond the scope of linear modeling.

The stacking analysis further supports this complementarity. In the stacked Meta-LGBM model, LGBM accounts for roughly 63.1% of the predictive contribution, while SVM contributes 36.9%. Moreover, the low correlation between the models' importance profiles ($r \approx 0.14$) shows that they rely on non-overlapping features. This minimal redundancy justifies the stacking approach, as it enables the integration of complementary linear and nonlinear signals to enhance overall volatility-prediction performance.

IV. Conclusion

This project presents a comprehensive framework for forecasting short-term (10-minute) realized volatility using high-frequency order-book and trade data. Through extensive feature engineering, the model captures dynamics across multiple temporal windows, addressing the asymmetric, heavy-tailed distributional characteristics of volatility. Employing a stacking architecture that integrates LGBM and SVM, the study demonstrates that combining nonlinear and linear learners yields superior predictive performance compared with either model alone or with naive benchmarks. The Meta-LGBM regressor leverages complementary information extracted by the base models, a result supported by the low correlation between their importance profiles and their reliance on non-overlapping predictors. This complementarity enhances robustness and improves RMSPE across cross-validation and out-of-fold evaluations.

The findings underscore the advantages of hybrid modeling strategies when dealing with complex, high-dimensional financial time series. The final stacked model not only delivers improved accuracy but also offers practical relevance for real-world trading applications, where reliable short-term volatility forecasts support pricing, hedging, and risk management decisions. Future work may extend this approach to incorporate additional microstructure signals, alternative neural architectures, or adaptive online-learning mechanisms suitable for rapidly evolving market conditions.

References

[1] 1st place solution – nearest neighbors. Kaggle. Available at: https://www.kaggle.com/competitions/optiver-realized-volatility-prediction/writeups/nyanp-1st-place-solution-nearest-neighbors.

[2] T. Dierckx, J. Davis, and W. Schoutens, "Using machine learning and alternative data to predict movements in market risk," *arXiv*, 2020. Available at: https://arxiv.org/abs/2009.07947.

[3] Z. Lin, "Modelling and forecasting the stock market volatility of SSE Composite Index using GARCH models," *Future Generation Computer Systems*, vol. 79, pp. 960–972, 2018.

[4] Optiver Realized Volatility Prediction. Kaggle Discussion. Available at: https://www.kaggle.com/competitions/optiver-realized-volatility-prediction/discussion/263321.

[5] Optiver Realized Volatility Prediction. Kaggle Overview. Available at: https://www.kaggle.com/competitions/optiver-realized-volatility-prediction/overview.
[6] Public 1398th → Private 24th Solution. Kaggle. Available at: https://www.kaggle.com/competitions/optiver-realized-volatility-prediction/writeups/republic-of-shigerunia-public-1398th-private-24th-.
[7] A. Vidal and W. Kristjanpoller, "Gold volatility prediction using a CNN-LSTM approach," *Expert Systems with Applications*, vol. 157, 2020.
[8] Y. Wang, H. Liu, Q. Guo, S. Xie, and X. Zhang, "Stock volatility prediction by hybrid neural network," *IEEE*, 2019.
[9] Q. Zhao, "A deep learning framework for predicting digital asset price movement from trade-by-trade data," *arXiv*, 2020. Available at: https://arxiv.org/abs/2010.07404.