SI670 Project Report

Chongdan Pan
pandapcd@umich.edu

Xin Ye
xinye@umich.edu

Fangzhe Li
fangzhel@umich.edu

December 10, 2022

Abstract

Bitcoin and cryptocurrency have been hot topics these years due to the rapid growth of the market value. Due to the lack of regulation, the market of cryptocurrency is very volatile and risky for investors. This project aims to construct a systematic way to predict the volatility of the Bitcoin market through the microstructure of the market. The data source is a high-frequency orderbook, which is a snapshot of all the orders listed on the exchange, and they're collected from Tardis, a major cryptocurrency data collector. Various machine learning models and features will be investigated and their performance will be evaluated by different metrics as well.

1 Introduction

Cryptocurrency such as Bitcoin and Ethereum is a novel digital currencies that have increasingly become a popular topic in the field of economy. They are "digital assets designed to work as a medium of exchange using cryptography to secure the transactions and to control the creation of additional units of the currency" [4]. More people have started investing in cryptocurrencies while they are also notable for being less stable and with stronger volatility [5]. There are two reasons for the rapid growth of the cryptocurrency market. First, since cryptocurrency is running in a decentralized, there is no restrictive regulation stopping investors from entering and exiting the market. Such freedom greatly attracts retail investors as well as financial institutions. Second, cryptocurrency is extremely volatile and opens 24/7 every year. The volatility and long trading hour imply that there are countless opportunities for traders to make money. However, with high returns, there must be high risk, which should never be ignored by investors. The new topic of forecasting cryptocurrencies' volatility has increasingly been an important topic in the finance area but it is not well-studied and fully explored. Therefore, this project aims to contribute to the prediction of the volatility of Bitcoin to not only help make a profit in the cryptocurrency market but also try to dig out more meaningful information from the market.

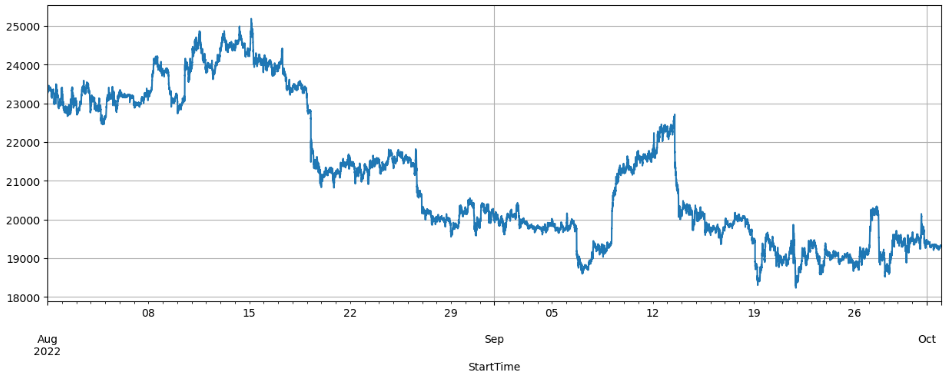
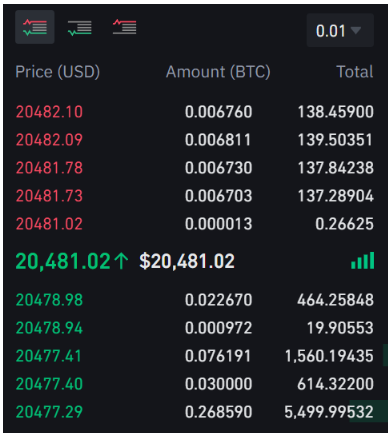


Figure 1: Bitcoin price

In this project, we're interested in studying the microstructure of the market of cryptocurrencies. Orderbook is the main source of the features we used, which is a snapshot of any market that stores all "buy" or "sell" orders information such as price and volume. Exchanges are usually made by matching orders from the price of the orderbook between buyer and seller. The orderbook data can provide insights into detailed trading information. Orderbook is generated extremely frequently because as long as anyone posts an order at the market, a new row of orderbook will be made. Therefore, the orderbook can be the detailed information source of the price movement. In this project, we used what we learned in SI 670, explored the bid price and ask price provided in orderbook, and constructed new predictors with feature engineering. By applying linear regression with ridge regularization and KernelPCA together, we successfully beat our baseline, which is Generalized AutoRegressive Conditional Heteroskedasticity(GARCH), a statistical model typically used to predicate the volatility based on the historical return. In addition, we explore other different models, including normal linear regression, XGBoost tree, and different time series models like LSTM and GRU. Finally, it turns out that our model can provide valuable insights through dynamic features from orderbook to explain Bitcoin price fluctuations.



Price (USD)	Amount (BTC)	Total
20482.10	0.006760	138.45900
20482.09	0.006811	139.50351
20481.78	0.006730	137.84238
20481.73	0.006703	137.28904
20481.02	0.000013	0.26625
20,481.02↑	$20,481.02	
20478.98	0.022670	464.25848
20478.94	0.000972	19.90553
20477.41	0.076191	1,560.19435
20477.40	0.030000	614.32200
20477.29	0.268590	5,499.99532

Figure 2: Sample live orderbook of Bitcoin

2 Related Work

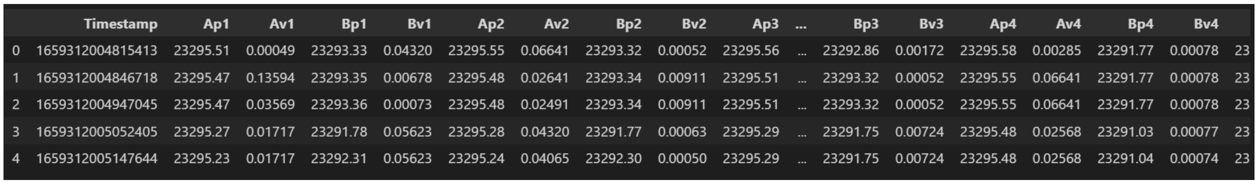
There are many previous studies that looked into this question based on different aspects of Bitcoin and ML techniques. For example, Rathan et al. [3] studied the prediction of Bitcoin price by feature selection of different machine learning techniques. They transformed orderbook data into features over time and then used the decision tree and linear regression model to develop predictions of the Bitcoin price for five days. Their experimental results compared the accuracy scores and found that compared to the decision tree, the linear regression showed a higher accuracy on price prediction. However, they didn't consider the potential use of existing forecasting models such as the Autoregressive Conditional Heteroscedasticity(ARCH) model in financial modeling and forecasting.

Guo.T and Antulov-Fantulin [1] looked into the orderbook data of Bitcoins and used this to make a short-term prediction of the exchange price fluctuations towards the United States dollar. In particular, they used the generalized autoregressive conditional heteroskedasticity model, the Beta-t-EGARCH model, and the structural time series model as baselines and derived a generative temporal mixture model based on the times series model with external features and temporal mixture model to adaptively learn volatility and the orderbook data. They also proved that the features of buy and sell orders significantly affect future high volatility. Guo et al. [2] also studied the problem of the Bitcoin short-term volatility forecasting based on the volatility history and orderbook data and proposed temporal mixture models which could successfully decipher the time-varying effect of orderbook features on volatility.

3 Datasets

3.1 Data Source

The data source for the project is Tardis, a fine-granularity cryptocurrency data collector, which provides all historical trading information of cryptocurrencies from different exchanges. This project focused on the data from Binance since it's the largest cryptocurrency exchange in the world. Since the orderbook dataset is too large with hundreds of fields and millions of records every day, the time range for the project will be passed 3 months, and only the top five ask and bid orders will be used. Bitcoin is chosen as the project's subject because it's the most popular cryptocurrency in the world with the largest market value. In summary, the dataset will have the following 21 fields as raw features:



	Timestamp	Ap1	Av1	Bp1	Bv1	Ap2	Av2	Bp2	Bv2	Ap3	...	Bp3	Bv3	Ap4	Av4	Bp4	Bv4	
0	1659312004815413	23295.51	0.00049	23293.33	0.04320	23295.55	0.06641	23293.32	0.00052	23295.56	...	23292.86	0.00172	23295.58	0.00285	23291.77	0.00078	23
1	1659312004846718	23295.47	0.13594	23293.35	0.00678	23295.48	0.02641	23293.34	0.00911	23295.51	...	23293.32	0.00052	23295.55	0.06641	23291.77	0.00078	23
2	1659312004947045	23295.47	0.03569	23293.36	0.00073	23295.48	0.02491	23293.34	0.00911	23295.51	...	23293.32	0.00052	23295.55	0.06641	23291.77	0.00078	23
3	1659312005052405	23295.27	0.01717	23291.78	0.05623	23295.28	0.04320	23291.77	0.00063	23295.29	...	23291.75	0.00724	23295.48	0.02568	23291.03	0.00077	23
4	1659312005147644	23295.23	0.01717	23292.31	0.05623	23295.24	0.04065	23292.30	0.00050	23295.29	...	23291.75	0.00724	23295.48	0.02568	23291.04	0.00074	23

Figure 3: Sample orderbook data of Bitcoin

- Timestamp: timestamp of the snapshot

- Bid Price[level]: The price of the top 5 buy orders with higher bid price.

- Bid Volume[level]: The volume of the top 5 buy orders with higher bid price.

- Ask Price[level]: The price of the top 5 sell orders with lower ask price.

- Ask Volume[level]: The volume of the top 5 sell orders with lower ask price.

And the final dataset we got is as follows: the training dataset is orderbook information for every 30 seconds from August 1st, 2022 to October 15th, 2022. And the test dataset is orderbook information for every 30 seconds from October 16th, 2022 to October 31st, 2022. We cut the data in this way because we want to make sure our train and test data have gone though different market conditions as much as possible, such as an extremely high peak in volatility.

4 Methods

4.1 Preprocessing

First, orderbook is not aligned with a specific frequency since people may post trade at any time, hence we need to resample the orderbook to reduce the size of the data and cut them into intervals with the same length, which is 30 seconds in our case. We choose 30 seconds because it will generate dataframes with the moderate size that our normal computers are able to manipulate.

After cutting the data, we can build volatility by ourselves, which is defined as the standard deviation of return within a specific time interval. Note that the volatility is calculated in different time horizons because the return within 30 seconds is too noisy and they usually follow a random walk. So we cut the price of Bitcoin by every minute, and calculate the return $R_t = \frac{P_{t+1}}{P_t} - 1$ of the t_{th} minute. Since the return is very small, we scale it by 100 times to avoid numeric issues during the calculation or backpropagation. Then we group the return by every 30 minutes and calculate the standard deviation as our label. In summary, our label is the standard deviation of all one-minute returns within 30 minutes, and we're using orderbook data with a frequency of 30s as features. Hence, with 30 minutes of data, we can calculate 60 returns and one volatility value as our label, and we can use 60 data points with 20 raw fields as our raw features.

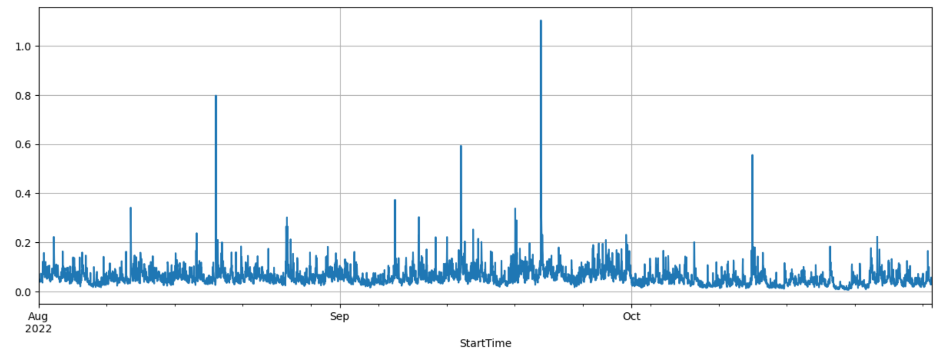


Figure 4: Bitcoin volatility

As shown in 4, the volatility is usually in the range between 0-0.2, but sometimes it'll become extremely high. Based on 5, it looks like the volatility follows a chi-square distribution with a low degree of freedom or gamma distribution. However, as we fit the data, it turns out that the P-value is very small, implying the volatility is not following any simple distribution.

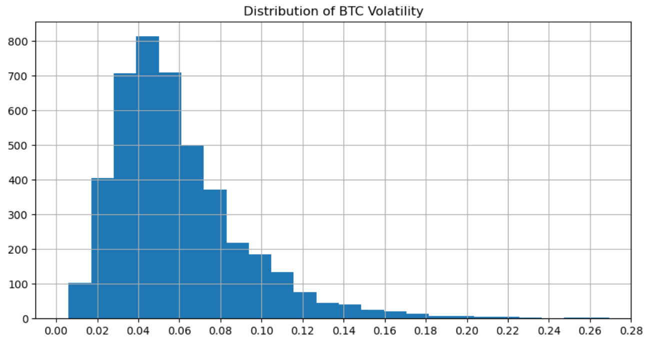


Figure 5: Histogram of Bitcoin volatility

4.2 Feature Construction

Besides the 20 raw features from the orderbook, we want to capture the structure of the orderbook and get more insights. Hence, we tried to imitate the behavior of human traders when they are looking at the orderbook and built the following features.

- Mid price: The average between the ask price 1 and bid price 1. This can be an easier and faster way to calculate the fair price of the crypto since both the buyer and seller need to pay the same amount of extra money to make the trade.

- Spread: The difference between the ask price and bid price and normalized by the mid price. This feature can be useful to evaluate the liquidity of the market. When the spread is larger, the cost for one to make a trader is higher, hence the liquidity is less. Low liquidity may lead to larger volatility people need to pay a lot for the price gap.

- Weighted Spread: Use the volume of each price as the weight to price to get the weighted difference between the ask price and bid price and normalized by the mid price. By taking more volume information into consideration, we can have a deep understanding of the spread.

- Volume Spread: The difference between the ask volume 1 and bid volume 1 and normalized by the sum of ask volume 1 and bid volume 1. This feature can show the strength of the price pressure. For example, when Av1 is very high, it's very hard to push the price up because you need to match a lot of orders placed at Ap1.

- Weighted Mid price: the weighted average ask prices and bid prices. Here we're calculating the mid price by taking the volume as the weight of the best bid and ask price. It is worth noting that we're using the volume of ask order as the weight of bid price and vice versa (e.g. Ap1 is multiplied by Bv1 as weight because the higher volume on bid order will potentially push the fair price up.

4.3 Dimension Reduction through PCA

As mentioned in the preprocessing section, we have 60 data points with 20 raw fields to predict one label, hence we have 1200 raw features in total, which is too much for some models because they'll lead to too many parameters and take a long time to fit. Hence, in some cases, we used PCA with a polynomial kernel to reduce the number of features.

4.4 Feature Normalization

We did standard normalization for features before they were fed to PCA because PCA is extremely sensitive to the scale of data. Since Bitcoin's trade volume and market price spread all have a different scale, we must standardize them so that they're treated equally by PCA and our models.

5 Models

5.1 Garch

The garch model is used as our baseline because it's a very popular model in the industry. It only takes historical return as features and uses the assumption of volatility clustering, which means if the volatility is high in t, then it'll probably be high as well at $t + 1$.

5.2 Linear Regression

Linear regression is selected due to its simplicity and interpretability. We're also using ridge regularization to avoid overfitting and PCA to reduce dimension here.

5.3 XGBoost

We're interested in the XGBoost tree model as well because it can give metrics of feature importance. In addition, it helps us to interpret the data and model better since we're able to plot the structure of the model.

5.4 Long short-term memory

Since the orderbook data are time series data, we think LSTM may be able to catch the relationship within the sequence. We're using both batch normalization and standard normalization since LSTM is very sensitive to the data scale as well.

6 Evaluation and Analysis

6.1 Evaluation Metrics

Since we're working on a regression task, MSE can be a very fair metric. We're interested in the R^2 as well because it reflects our models' ability to interpret the data. It turns out linear regression with strong ridge regression has the highest R^2 while LSTM has the lowest MSE.

Model	MSE	R^2
Garch	9.4E-4	0.05
Ridge with raw orderbook	8.9E-4	0.1
Ridge with PCA and our features	7.8E-4	0.21
XGBoost	9.3E-4	0.064
LSTM	8.8E-4	0.1
LSTM with dropout and batch normalization	1.8E-3	-0.83

6.2 Analysis and Discussion

6.2.1 Garch

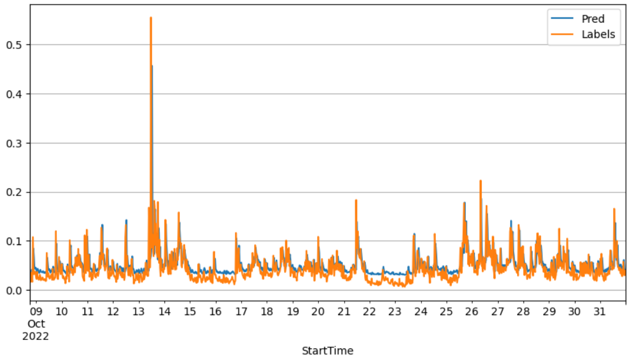


Figure 6: Prediction of Garch

As shown in the Fig. 6, our baseline is good at predicting the peak of the volatility thanks to its assumption of volatility clustering, but its lower bound is too high, leading to a high error.

6.2.2 Ridge

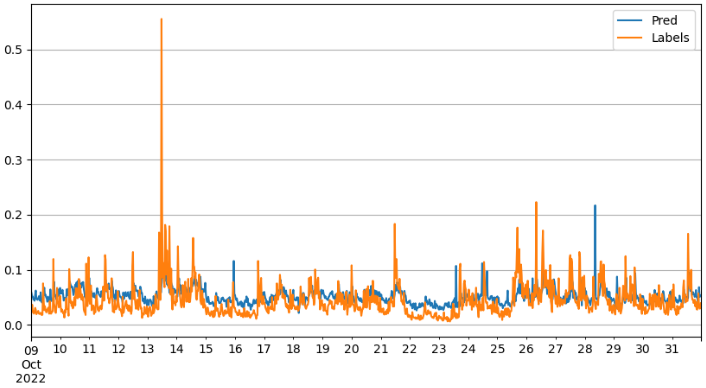


Figure 7: Prediction of Ridge with Orderbook features

Fig. 7 shows that simple linear regression can achieve a better result than Garch even though it can't capture the high volatility. Its supremacy implies that the orderbook's features contain a lot of information for prediction.

After we take our own features into consideration and reduce the dimension with PCA, the result is much better. Compared to Fig. 7, the prediction in Fig. 8 has a less wrong prediction of peak and its normal prediction is closer to the labels.

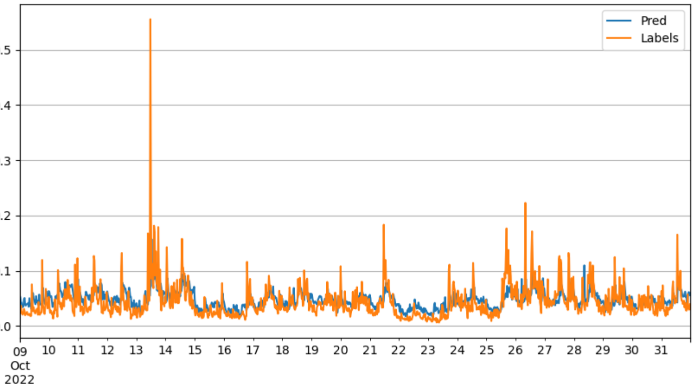


Figure 8: Prediction of Ridge with our features and PCA

.

6.2.3 XGBoost

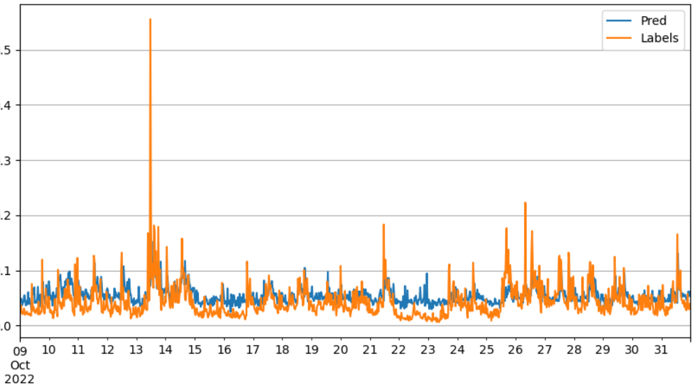


Figure 9: Prediction of XGBoost

Surprisingly, XGBoost can't achieve an ideal performance for prediction even though its ensemble methods are famous for solving underfitting. The worse performance implies that XGBoost is overfitting here, and our best model is actually able to exploit all linear information within the features we have.

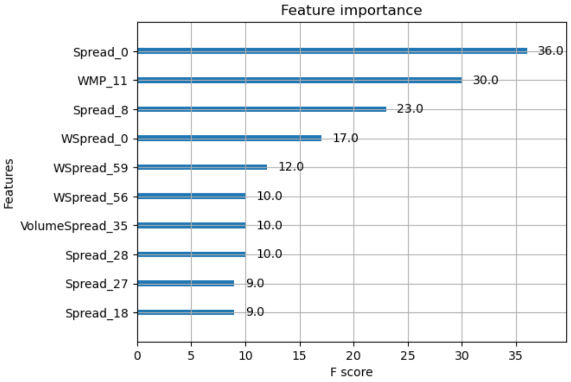


Figure 10: XGBoost Feature Importance

With that being said, Fig. 10 shows that spread and weighted mid price plays a significant role in volatility, implying the liquidity and momentum effect in the microstructure of the market.

6.2.4 LSTM

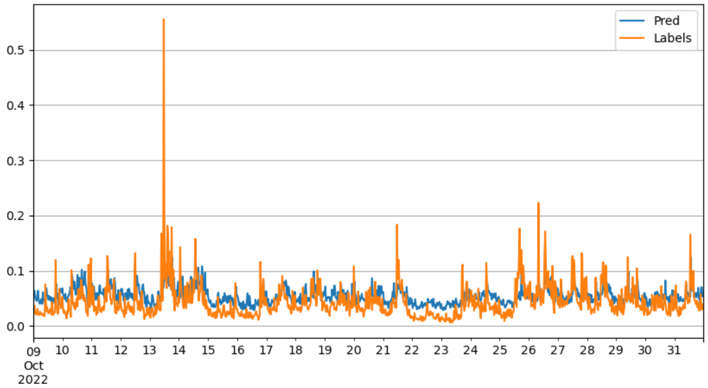


Figure 11: Prediction of LSTM

We also tried LSTM as time-series neuron-network model. LSTM turns out to be as good as our ridge model with PCA. We guess it's probably because the sequence length is not ideal and the data set is too small to train an LSTM neuron network. We believe if we can handle more data, LSTM should be more robust.

7 Conclusion

From this project, we can see that almost all our models can beat the baseline, which shows that with the information included in the orderbook, we can easily beat the prediction of the baseline model only based on return rate information. However, since only the Garch model can successfully predict the volatility peak, in the future, we may use the output of the Garch's prediction as another feature.

Among the four machine-learning-based models we implemented here, our Ridge regression model with more features gives the best performance. However, our trial of other models gives us the previous and insightful experience of how different models work and how to exploit them even when they're not performing well.

8 References

[1] Guo, T., & Antulov-Fantulin, N. (2018). Predicting short-term Bitcoin price fluctuations from buy and sell orders. arXiv preprint arXiv:1802.04065.

[2] Guo, T., Bifet, A., & Antulov-Fantulin, N. (2018, November). Bitcoin volatility forecasting with a glimpse into buy and sell orders. In 2018 IEEE international conference on data mining (ICDM) (pp. 989-994). IEEE.

[3] Rathan, K., Sai, S. V., & Manikanta, T. S. (2019, April). Crypto-currency price prediction using decision tree and regression techniques. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 190-194). IEEE.

[4] Tsai, Wei-Tek, R. Blower, Y. Zhu and L. Yu, "A system view of financial blockchains," in 2016 IEEE Symposium on, 2016.

[5] J. Chu, C. Stephen, N. Saralees and O. Joerg, "GARCH Modelling of Cryptocurrencies," Journal of Risk and Financial Management, vol. 10, no. 4, p. 17, 2017.

[6] Peter R. Hansen and Asger Lunde. 2005. A forecast comparison of volatility models: does anything beat a GARCH(1, 1)? Journal of Applied Econometrics 20, 7 (2005), 873–889.