MS&E 448 - Crypto Exchange Predictions

Paul Boehringer

June 9, 2020

1 Strategy Description

This paper focuses on the XBTUSD crypto currency pair on the BitMEX exchange. The XBTUSD pair is designed to track the price of Bitcoin however one does not actually own Bitcoin after purchasing an XBTUSD contract as it is a perpetual swap that is funded every 8 hours. BitMEX's daily volumne on XBTUSD is $1.82 Billion which they claim to be 1500% of the liquidity of Bitcoin on any exchange where the actual coin is traded. This liquidity is extremely attractive to traders as it makes execution more consistent. Due to this liquidity there has been a rise of trading volume in swaps for other coins, while the methods used in this paper could apply to any contract on the BitMEX exchange a lack of data only allows for the analysis to be done on the XBTUSD contract.

The strategy takes BitMEX's public trade by trade data set which is available daily for the past 3 years. First, summary statistics such as the number of buys, sells, plus ticks, or minus ticks each minute's interval are found. Then an LSTM takes the prior 57 minutes of history and makes predictions for the next 3 minutes about when the minimum and maximum prices will occur along with that they will be relative to the prior minute's close. These predictions are used to make trading decisions. While the initial predictions are made for a 3 minute holding time the final strategy is allows for holding as long as possible as the sale is determined by a stop loss which is continuously edited to be 15 ticks (or pennies) below the current value of the contract.

The returns of the strategy are heavily dependent on the market environment and how long the stop loss order has been in before it is executed. Expected annual returns range between 10% and 53% with a corresponding Sharpe ratio of 0.67 and 2.72 with the most likely value being right between the worst and best case scenarios.

The most significant unknown on the strategy is whether execution would have an extreme impact on BitMEX's order book. This becomes highly relevant to the strategy's profitability as profit is heavily based on trading fees which change based on a trader's 30 day volume. If the strategy maintains stability for a $10 million trading volume over 30 days, then the returns would drastically increase as trading fees drop by an order of magnitude (factor of 10). Above this trading volume expected returns jump to 340% annually and the expected Sharpe ratio becomes 6.7. However, it is believed that this kind of trading volume would have drastic effects on the exchange eroding the profitability of the algorithm.

2 Data & Investment Universe Selection

Note: These two sections are combined as the data set heavily effected the choice of assets to trade

The dataset is available on the BitMEX website [1]. There is a row for every trade executed on the BitMEX exchange. This consists of an average 1.6 million trades per day over the last 30 days (as of 6/8/2020). Of these, approximately 600,000 trades are XBTUSD while the others are split between a list of 13 other swaps. Notably, Etherium (ETHUSD) has been on the rise as it makes up a hefty 250,000 of this 1.6 million daily trade count. While the alt-coin volume went up significantly over the past year, only XBTUSD has consistent enough data to make trading decisions.

Also, due to the lack of alt-coin data an additional data set had to be gathered from CoinBase [2]. This was Open, High, Low, Close (OHLC) data for Ethereum and Litecoin broken down into minute intervals.

2.1 Input Feature Creation

Due to the size of the data set over the past 3 years, the BitMEX files were scanned in 5 day intervals. During this scan the following summary statistics of each minute were recorded.

- Volume Weighted Average Price
- Number of [zero minus ticks, minus ticks, plus ticks, zero plus ticks]
- Volume in XBTUSD (buy and sell)
- Minimum Buy & Sell
- Maximum Buy & Sell

Making for 11 initial numbers to describe the behavior of XBTUSD over each minute.

Note that no intra-interval features maybe calculated during this file scan due to continuity issues. Once all the data is processed in 5 day batches the intra-interval features which need overlap from the last interval in one 5 day batch to the first interval in the next batch are created. This is done by taking the minute interval summary which has been created and calculating the relative rate of change for various features of the interval statistics. Once the formulation of one of these relative change statistics is clear, the meaning of the others may be inferred based on their name alone. To see this take take the intra-interval feature *MinuteMinBuy_V_VWAP*'s as an example which is calculated by:

$$\text{MinuteMinBuy\_V\_VWAP}_t = \frac{MinuteMinBuy_t - VWAP_{t-1}}{VWAP_{t-1}}$$

That is, *MinuteMinBuy_V_VWAP* is relative value of the minimum buy price in the current interval compared to the VWAP during the prior minute. If a metric's value is compared to itself then it is labeled as that metric's return. For example *MinuteMinBuy_return* being the relative value of the minimum purchase price of XBTUSD in one interval to the next.

Note the features that consider volume and count statistics are included in both inter-interval form (i.e. their actual counts) along with relative change of the value from one period to the next (i.e. its intra-interval statistic). The list of the final features that are fed into the LSTM can be found in the appendix.

2.2 Output Target Creation

The model will use forecasts of price characteristics for 3 minutes into the future to make trading decisions. These predictions describe the maximum and minimum for both buy and sell prices along with when these will occur in that 3 minute interval. The main three features of focus are *min_buy_price, max_sell_price, min_buy_to_max_sell_when.* In simple terms the minimum purchase and maximum sell price of XBTUSD and an estimate of the time between these two events. There are an additional 7 target values for this three minute interval. The whole list of targets is included in the appendix.

3 Modeling

Before covering how the inputs will be used to make predictions for the outputs it is important to consider the flow of the LSTM and how it is used in the decision making process, this is shown in *figure 1* below.

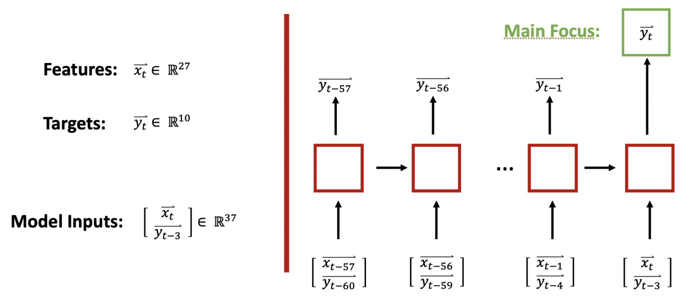
Features: $\vec{x_t} \in \mathbb{R}^{27}$
Targets: $\vec{y_t} \in \mathbb{R}^{10}$
Model Inputs: $\begin{bmatrix} \vec{x_t} \\ \vec{y_{t-3}} \end{bmatrix} \in \mathbb{R}^{37}$
Main Focus:
$\vec{y_t}$
$\vec{y_{t-57}}$
$\vec{y_{t-56}}$
$\vec{y_{t-1}}$
...
$\begin{bmatrix} \vec{x_{t-57}} \\ \vec{y_{t-60}} \end{bmatrix}$
$\begin{bmatrix} \vec{x_{t-56}} \\ \vec{y_{t-59}} \end{bmatrix}$
$\begin{bmatrix} \vec{x_{t-1}} \\ \vec{y_{t-4}} \end{bmatrix}$
$\begin{bmatrix} \vec{x_t} \\ \vec{y_{t-3}} \end{bmatrix}$

Figure 1: LSTM Flow of Inputs and Outputs

Statistics from the past 57 minutes of XBTUSD along with some pricing information from Ethereum and Litecoin are used as inputs. When an LSTM is making these predictions it became clear that the prior values of the outputs also have an effect on the future. Because the actual values of the outputs are known after the 3 minutes into the future have passed, x_t and y_{t-3} can be appended together to form an input to the LSTM which will be totally known at the time of prediction making process. It is important to give the model this feedback otherwise it will make blind predictions about characteristics of the price without considering what has actually happened.

3.1 Why an LSTM

LSTM stands for Long Short Term Memory which is a recurrent neural network (RNN) structure. RNNs are meant to process sequences of data where past events have some effect on the future. An LSTM is composed of identical cells which consist of an input gate, output gate, and forget gate. At each time step the cell has the capability to remember past features of events which could become relevant at an arbitrary point in the future depending on the current state. This is done using a mechanism called attention which is stored in the hidden state that is updated depending upon what is fed through the network [3].

3.2 Training Logistics

The only prediction which is relevant is the final prediction of the LSTM. The earlier ones are not as important for two reasons. First, when the model is implemented in real time the earlier events have already happened. Second, the LSTM is *'learning'* the current market regime as each prior minute inteval is fed into the network the hidden state of the network is updated.

For the model to figure out what kind of market it is in it will need to have exposure to many regimes. To do this it is best to train over all of the data available. However this poses a logistical issue due to computation resources. That is because the final prediction is the most the only metric that will be used to gauge performance. Each final prediction the network makes requires 57 forward passes through the network meaning that the input shape of X to the LSTM if all the data were to be used is a tensor of shape [#_observations, #_time_steps, dimension of x_t] So training on the whole data set would mean X has shape [1,576,000, 57, 37] and the corresponding Y, or the outputs, would have shape [1,576,000, 57, 10]. This right away becomes too much to train on without using many GPU's with parallel computation. To help alleviate the problem a random subset of 5% of all of the data available is selected from the beginning to the current period minus the previous 2 weeks for validation reasons. The network is then trained for 15 epochs on all of this data.

Then another 15 epochs are run on the network, this time using most current 1.5 months. Splitting is done using SciKit Learn's time series split where 1 week of this data is selected for validation and the other 5 weeks are used for training. This is more clearly illustrated in *figure 2* below.

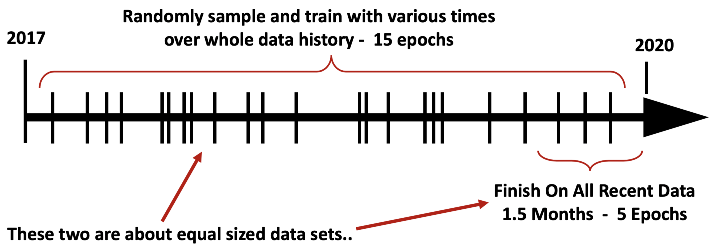
Randomly sample and train with various times over whole data history - 15 epochs
2017
2020
Finish On All Recent Data 1.5 Months - 5 Epochs
These two are about equal sized data sets..

Figure 2: Sampling Method for Regime Exposure

So first the model is trained on the whole timeline shown above. Then it is trained 6 different times where a different week is selected from the 1.5 month period for validation. Performance was fairly consistent regardless of which of the 6 weeks were used for validation.

4 Alpha Model

The underlying idea of this model is incredibly simple, predict what the lowest buy price and the highest sell price over the next 3 minutes will be along with when they occur. The idea is the epitome of a buy low sell high mentality. While the idea itself is simple the complication comes when making these predictions.

A significant part of the intuition for the predictions of this model came from a paper which used historical order book data to find universal price formation patterns in the NYSE [4]. What is notable is that this paper works only with order book data and found that using enough historical data they can make state of the art (SOTA) predictions for whether the next tick of an order book will go up and down. This was interesting because the authors used data which had nothing to do with the fundamentals of a stock and it still outperformed models which brought information that is classically considered to be tied to the price of a stock into account.

Cryptocurrencies seemd like a great application for this kind of numerical technical analysis as there is no fundamental driver for the price of Bitcoin, Ethereum, or any other cryptocurrency. If this project was able to find underlying patterns of prices for equities which have fundamental value then it seemed natural to apply the same sort of analysis to something without it. Notably the idea of using order book level data for Bitcoin on the exchange was toyed with, however this data would require orders of magnitude more compute power and storage and hardware limits were already being pushed so this was put off as an idea for later.

While there are no conditioning variables specifically this idea is somewhat integrated into the model by first training across the whole time period of available data. This will build a prediction surface which is more robust as it has seen a wide range of the asset's behavior. Then it can refine its behavior based on the current regime as mentioned in the modeling section by fine tuning on more current data. This methodology clearly added value. When training was only done on the most recent data the network's predictions were good on the training set but tended to be near zero when making predictions on the validation set. This tends to happen when the model can't pick up much meaning so it makes predictions near 0 to minimize error. However, when the initial training is done on the asset's whole data set the predictions spread out much more noticeably. Below in *figure 3* is a plot of an experiment where the first network is trained only on the most recent dense data set compared to training over the whole history.

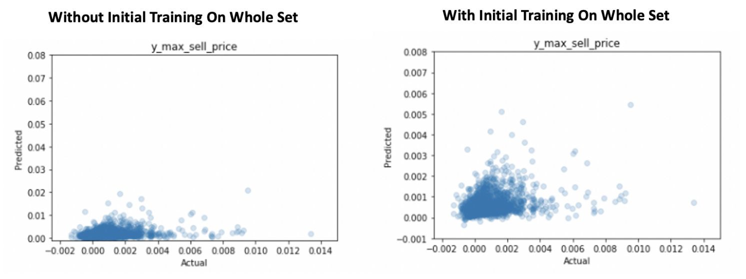

Figure 3: Effectiveness of Training Method for Regime Exposure

This became a key factor in the performance of the model as the timing predictions were not as good as initially hoped so the model relied more heavily on relative change of prices for predictions. While the network has very little ability of predicting when the low and high prices will occur or the time between the two it does a slightly better job of predicting whether the the low will happen before the high in extreme cases. This will be covered more in the next section on portfolio construction.

5 Portfolio Construction

While the network does a poor job of predicting exact timing it tends to be accurate in predicting whether the low comes before the high at the extremes. Below in *figure 4* is the predictions of the time between the high and the low.

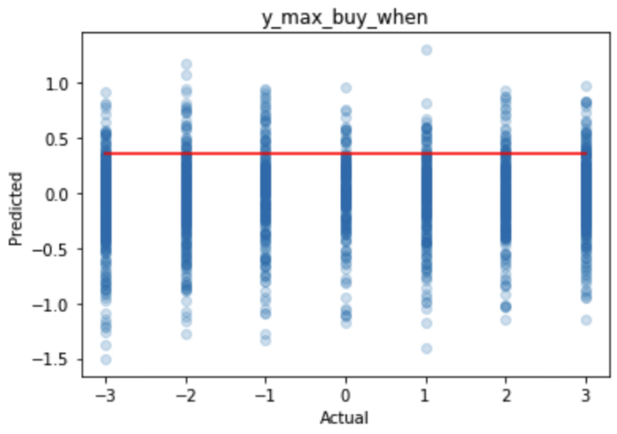

Figure 4: Predictions of Time Between Min Buy and Max Sell

Note the red line on the plot which represents the decision threshold for a trade. Only predictions which lie above this line will end in a limit order being placed. This is an arbitrary threshold that was found to work well. That is that the predicted time between the low and the high is greater than 0.35. The accuracy of predicting the low before the high is only 62% which is not too great considering one would expect a 50/50 split from random guessing. However using some simple mechanical order placement rules this information can become profitable quite quickly.

The steps are as follows:

```
Make pricing predictions at the end of each minute
if predicted timing of min to max $\geq$ 0.35 then
    place limit purchase order for 50% of portfolio currently allocated to
      USD at 0.933 times the value of the minimum predicted price in
      that time interval
    if purchase order executes in less than 2 minutes then
        set stop loss for $0.15 loss per XBTUSD contract on that trade
        while stop loss order does not execute do
            if XBTUSD price $\geq$ stop loss price + $0.15 then
                update stop loss order to XBTUSD price - $0.14
            end
        end
    end
    else
        cancel purchase limit order
    end
end
```

The algorithm is willing to increase exposure to XBTUSD up to 100% of the portfolio. However it will never hit 100% in the same manner that if one walks half the distance to a wall they will never hit the wall. This is because every time the predicted criteria for purchasing is met a limit order is placed for 50% of the portfolio's current allocation to cash. While the algorithm will never be all in on XBTUSD it will spend much of the time with 100% cash due to the nature of the constant stop loss orders being placed. The reasoning behind this allocation method is that the algorithm will always allow for an increase of exposure if it seems that the asset will increase in value, simply if it thinks Bitcoin will keep going up it can still increase exposure. A few alternates to 50% were experimented with (mainly 40% and 60%). None of them stuck out more than the others. Performance was based on trend: if the price was heading down 40% did the best and vise-versa for 60% allocation. Admittedly, much more experimentation could be done here.

Overall no formal optimizer was used, but experimenting with hyper parameter tuning by changing the threshold for required predicted time between the low and high price from 0.35 was also done. This threshold was not always the best value but it did yield the most consistent results which is why it was chosen. Finally onto the first set of portfolio performance in *figure 5* below.

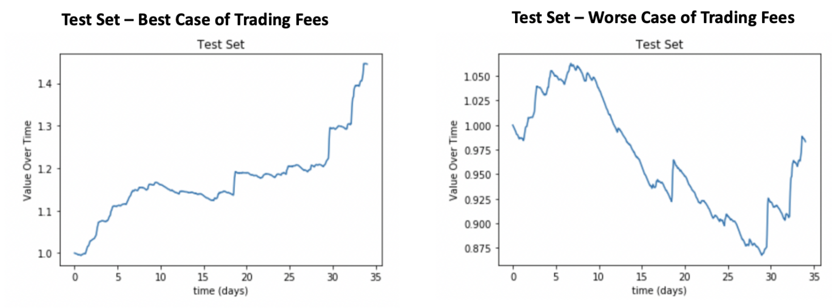

Figure 5: Predictions of Time Between Min Buy and Max Sell

This is the performance of the most recent 35 days available on the data set. Note, the difference between the best and worst case of trading fees depends on whether or not a trade is rewarded a market maker's fee of 0.025% for providing liquidity. This will happen if a stop loss order is in the order book for long enough. While there will never be a fee paid to BitMEX for providing liquidity by executing a sell order using a limit they will pay you if the order is in the book long enough. The exchange provides traders with this small rebate as an incentive for providing liquidity as the exchange focuses primarily on options trading so they have an interest in guaranteeing liquidity for the market. Creating a detailed back tester which could determine whether a market maker's reward would be provided was attempted, however time ran out before seeing this to completion. Regardless assuming all trades execute real performance will be somewhere between these two plots.

The same model which is shown on the test set was ran on the prior year's worth of data (except for the portion from the dense set which the model is trained on). From there the best two week run was selected to show the potential for this strategy to yield steady returns with minimum down draw.

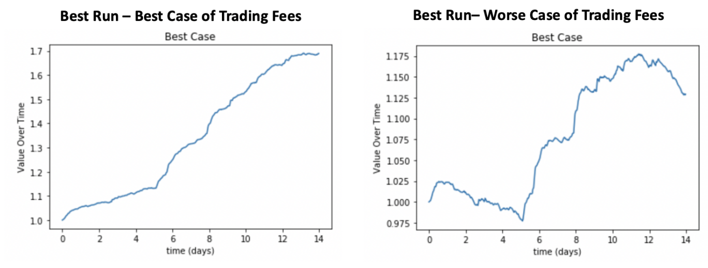

Figure 6: Predictions of Time Between Min Buy and Max Sell

6 Risk Management Philosophy

Risk is managed through the use of stop loss orders. The fact that the there is always a stop loss of $0.15 from the current price of an XBTUSD contract means that downside risk is protected. What is uncertain is what would happen during black swan like events where the asset experiences a flash crash. This is the most concerning issue with the strategy. Through out the back testing of this strategy it was assumed that these stop loss orders always hit 2 ticks (or pennies) below the stop loss price as a conservative "guess" of what these extreme events will do to the strategy on average. However there is no evidence to support that this is a reasonable assumption. However given the short amount of holding time this maybe a fair assumption.

As mentioned in the strategy overview in the beginning of the paper the Sharpe ratio for the strategy on the test set case is 0.67 for in the worst case of trading fees and 2.72 in the best case. However, it should be noted that the best case of trading fees on the best run has an absurd Sharpe ratio of 380, this plummets to a Sharpe ratio of 15 when the worst case of trading fees are considered for the best run.

Overall the biggest risk of this strategy will be what happens to profitability when black swans or flash crashes occur as the behavior can not be modeled without tick by tick order book level data.

7 Execution Discussion

BitMEX is a very friendly exchange with regard to execution. They have an API which is recognized to be one of the best of any crypto exchange. Further they boast a huge level of liquidity. At the time of writing the strategy summary (section 1) the 24 hour liquidity was \$1.82 billion, however over the course of 2 days and checking back in liquidity is now at \$1.91 billion for the past 24 hours.

The back testing on the validation data set led to 770 trades being executed over 35 days. It seems reasonable that this strategy would not have too much of an effect an exchange with such a large volume if the trading volume for the algorithm during these 35 days is kept below \$1,000,000 or about \$1,200 per trade.

Another option which was not considered in this paper is the ability to trade on with leverage. BitMEX provides the capability to trade at 100 times margin with fairly reasonable fees. While this may drastically increase profits it comes with the risk of more severe down draws. This should not be experimented with until the un-leveraged strategy has been in real time operation for a few severe down draws to see how trades end up executing during those times. However, this opens new doors for the algorithm's implementation. Instead of putting in 50% of the cash balance each time an interval meets the desired criteria the whole porfolio value could be purchased and held instead. For example if 5 intervals meet the criteria in a run where no stop loss orders are hit then the portfolio would be at 5x leverage instead of having $1 - 0.5^5$ or about 96.8% of the portfolio in XBTUSD.

I did make an attempt to build a back tester, however there is no easy way of telling how long a stop loss order will sit an the book. This is part of the reason I decided to assume each limit order is executed \$0.02 below the actual price, because there may be others who placed a limit to sell at the price before the algorithm places the order. In this case the price received is driven down further. While 2 pennies per contract may be a large loss for many equities its important to note that these 2 pennies or ticks is only 0.02% of the price of an XBTUSD contract at the time of writing (as Bitcoin is hovering around \$9,700).

8 Retrospective Discussion

Overall, this is the first project I have worked on with such a large data set that forced me to take extreme caution during the pre-processing steps. Scanning all the trades over the whole 3 year period and building summary statistics in a memory efficient manner proved to be much easier said than done while making sure the code ran at a reasonable speed. I have never found the need to use the garbage collector in python until now. While I'm sure my naive use of it at the end of every 5 day scan was not optimal it did half the time of the file scan.

The other thing that was very cool to see work was training on all of the data then fine tuning the model to the current regime by tuning on the most recent data. This was notable because it is something that is common practice in natural language processing. It was nice to see this methodology work on a totally different application.

Finally, I was very upset with the accuracy of timing predictions from my model, however I found it cool how adding some mechanical order rules turned what seemed to be a hopeless model into something that has promising performance.

References

[1] BitMEX. *BitMEX Trade by Trade Data*, 2020.

[2] CoinBase. *Coin Base API*, 2020.

[3] Shreya Ghelani. *Breaking BERT Down - A Complete Breakdown of the Latest Milestone in NLP*, Jul 26, 2019.

[4] Rama Cont Justin Sirignano. Universal features of price formation in financial markets: perspectives from deep learning. *Journal of Quantitative Finance*, 19(9):1449–1459, 2019.

A Input Features

Inter-Interval Features
'MinusTickCount', 'ZeroMinusTickCount', 'ZeroPlusTickCount', 'PlusTickCount', 'MinuteBuyVol', 'MinuteSellVol'

Intra-Interval Features

'vwap_returns', 'MinusTickCount_returns', 'ZeroMinusTickCount_returns', 'ZeroPlusTickCount_returns', 'PlusTickCount_returns', 'MinuteMinBuy_returns', 'MinuteMinSell_returns', 'MinuteMaxBuy_returns', 'MinuteMaxSell_returns', 'MinuteBuyVol_returns', 'MinuteSellVol_returns', 'MinuteMinBuy_V_vwap','MinuteMinSell_V_vwap', 'MinuteMaxBuy_V', 'MinuteMaxSell_V_vwap', 'ethLow_V_XBTMinuteMin', 'ethHigh_V_XBTMinuteMax', 'ethClose_V_XBTVWAP', 'ltcLow_V_XBTMinuteMin', 'ltcHigh_V_XBTMinuteMax', 'ltcClose_V_XBTVWAP'

Target Setbacks
'y_max_sell_price', 'y_max_buy_price', 'y_min_sell_price', 'y_min_buy_price', 'y_max_sell_when',

'y_max_buy_when', 'y_min_sell_when', 'y_min_buy_when', 'y_min_buy_to_max_sell_price', 'y_min_buy_to_max_sell_when'

B Output Targets

'y_max_sell_price', 'y_max_buy_price', 'y_min_sell_price', 'y_min_buy_price'

Timing Targets 'y_max_sell_when', 'y_max_buy_when', 'y_min_sell_when', 'y_min_buy_when'

Difference Targets 'y_min_buy_to_max_sell_price', 'y_min_buy_to_max_sell_when'
buy_to_max_sell_when'