MS&E 448 - Project Report
Cryptocurrencies Trading Strategy based on Sentiment Analysis

Arnaud Autef **Catherine Gu** **Olivier Pham** **Charbel Saad** **Benoit Zhou**

Abstract

Cryptocurrencies are an appealing asset class. The overall market capitalization has been growing rapidly while the barrier to entry for trading is very low. However, cryptocurrency prices are volatile and speculative, making return predictions challenging using traditional financial tools. In this project, we leverage Sentiment Analysis data to explore trading strategies and construct alternative alphas on cryptocurrencies. This work is motivated by the observation that a substantial amount of interests and developments in the crypto universe happens through modern social media. We thereby delve into data from Google Trends and Twitter to construct predictors for asset returns using high-frequency Level I data from Coinbase exchange. A thorough data exploratory analysis on tweets provide us with tools to build a successful trading strategy on BTC-USD with an ex-ante Sharpe ratio of 0.381 with 5 bps of (linear) trading costs. To further our research on algorithmic trading, we demonstrate a strategy concept following high frequency trading on Litecoin returns based on Bitcoin price movements to illustrate arbitrage opportunities that may be reproducible across exchanges. By efficiently combining high-frequency sentiment analysis with market-based predictors, such a strategy can carry multi-level investment horizon in extracting interesting alphas that could be successfully implemented in the future for cryptocurrency trading.

1 Introduction

The first decentralized cryptocurrency, Bitcoin, was created in 2009. Ten years forward, there are now over 2,200 cryptocurrencies and tokens being actively traded at over 19,000 different exchanges across the world, with a staggering combined market capitalization of $250 billion[1]. The barrier to entry to this market is fairly low, and similar to trading equities, it does not require expensive data infrastructure for both retail investors and institutional investors to become active. Moreover, the rich trading universe provides a unique opportunity to combine financial portfolio construction techniques with creative feature engineering to this new asset class as we search for alternative alpha sources.

One of the salient characters of the crypto asset class is its volatility. Volatility in cryptocurrencies is partly driven by the fact that crypto assets lack fundamentals for support - indeed many cryptocurrencies being actively traded today (including Bitcoin) simply cannot be evaluated by their real economic values using any traditional financial modeling tools we are familiar with. This in turn gives way to speculators who drive the market for trading purpose as opposed to investment purpose. Combining this with the fact that many cryptocurrencies trade in low volume and have liquidity constraints - since Bitcoin alone occupies over half of the total crypto market cap - the challenge of predicting returns on crypto assets is well-known. This observation is the starting point of our strategy. Our goal is to combine traditional market-based algorithmic trading techniques with alternative alpha construction from sentiment analysis data. By analyzing timely sentiments from

[1]Coinbase Data as of June 5, 2019

social media data and crypto forums at high frequency, we are better at capturing the speculative and thereby volatile nature of cryptocurrencies and to efficiently trade on those markets.

Our main fund is a long/short crypto portfolio trading the currency pair Bitcoin against US Dollars (BTC-USD). We have selected this instrument to trade because BTC-USD is the most liquid crypto market, with also the highest trading volumes. Likewise, sentiment data available for Bitcoin has the highest quality in terms of volume and relevancy. We have carefully designed sentiment-based signals by experimenting with several social media platforms for data scraping as well as varied sentiment analysis packages, which we will examine in detail. We focus on short-term trading strategies with a trading horizon that ranges from hourly to a maximum of one day. We construct daily signals from well-designed sentiment scores and apply filtering techniques to condition our predicted returns in order to obtain high conviction trades for which the portfolio algorithm will implement. Strategies derived from these predictors are evaluated through backtests. Our portfolio construction closely imitates real trading environment by introducing alpha threshold that parameterizes portfolio turnover rate and by introducing trading cost. We assume negligible slippage cost due to market impact. The ex-post Sharpe ratio on our portfolio is 1.708 assuming 5 basis points (bps) trading cost and 2.275 without trading cost, respectively, for the one-year period between 2018 and 2019. Training the model again on the current year of 2019, our Sharpe ratios are 0.567 without trading cost and 0.381 with 5bps trading cost.

Given the vast number of tradable instruments available for cryptocurrencies, we also consider combining strategies with market-based indicators to improve portfolio performance. This includes incorporating more granular types of market data. Most of our data are Level I data that incorporates pricing information. We wish to include data that reflect the trading dynamics, such as the most recently completed trades or even Level II data that incorporates the whole order book in order to extract better market-based alpha for our portfolio. We also have a proof of concept strategy for Litecoin as an extension based on high frequency statistical arbitrage. The concept helps to illustrate opportunities for arbitrage trading across different exchanges - a long term strategy that we could combine with sentiment analysis.

The paper is organised as follows: Section 2 explains the **Data Collection** process. Section 3 discusses our **Work on Tweets**. Section 4 discusses our **Work on Google Trends**. Section 5 presents the **Results** of our main trading strategies. Section 6 extends by **Coupling Sentiment Analysis Alpha with Market-based Predictors**. Section 7 summarizes our research work in **Conclusion**.

2 Data collection

Our data Universe is cryptocurrencies open/close/low/high/volume prices Level I data, and social media and news data, with a temporal scope of 2018 and 2019. More precisely, for the cryptocurrency prices, we pull the data from the Coinbase exchange, and use its API to retrieve historical and real-time prices with minute granularity of the currencies we are interested in. As for our social media and news data, we are using Twitter and Google Trends.

Google Trends is a website that analyzes the popularity of top search queries and keywords. However, the raw keyword level data spreadsheets from Google Trends need some preprocessing: we first download yearly data at weekly granularity, then weekly data at hourly granularity, and finally collate the two of them after normalization to obtain continuous data at hourly granularity.

The Twitter API does not allow us to scrape data further than one week old, hence is not useful for our project as we require historical data. Instead, we use the python project GetOldTweets[2] to bypass the official Twitter API and scrape older tweets. However, it retrieves all the tweets for 24 hours between 4pm and 4pm in chronological order, which is too many tweets for us to handle. We therefore limit the data volume by only using GetOldTweets to retrieve data between 3 and 4 pm, which approximately represents 1000 tweets. It is noteworthy that even though we could not use the Twitter API for our project, it would be a great resource to get real-time data and continue working on more advanced models in the future.

[2]https://github.com/Jefferson-Henrique/GetOldTweets-python

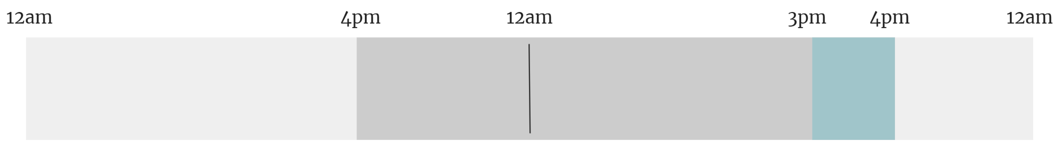


Figure 1: Data retrieved by GetOldTweets (dark gray) and data we are working with (turquoise)

3 Work on Tweets

3.1 Sentiment Analysis Features

NLP Analyzer To construct our sentiment analysis (SA) features, we use natural language processing models called *sentiment analyzers* which, for a given text returns a sentiment polarity score that can be:

- positive: enthusiastic / positive tone dominating the text
- negative: pessimistic / negative tone dominating the text
- neutral: no dominant tone in the text

Social media data are intrinsically complicated; it can be challenging to infer an accurate aggregated sentiment from a disparate source of tweets and posts. In selecting the package for us to use that can produce the most reliable results on the social media data that we have scraped, we randomly sampled 1,000 tweets and examined the distribution of sentiment scores from three different packages that are popular amongst practitioners.

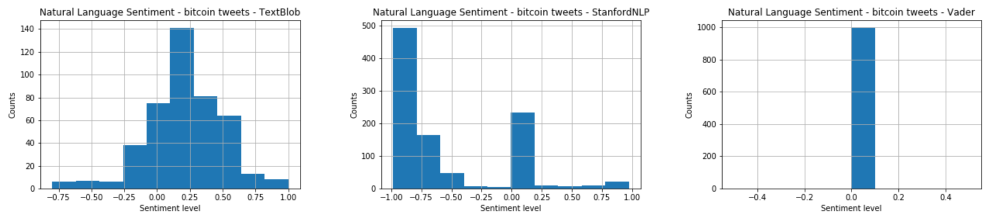


Figure 2: Sentiment Score Distribution with TextBlob[3], StanfordNLP[4], Vader[5]

Based on the analysis, we choose *TextBlob* for our NLP analyzer which is built on a NaiveBayes classifier. Its sentiment score distribution seems to suggest more coherent results and less noise when sampling a combination of negative and positive random tweets. In comparison, *StanfordNLP* seems to produce an overwhelmingly negatively skewed sentiment distribution, a result which is also confirmed when we constructed manually various sentences for testing; *Vader* sentiment analyzer seems to be very insensitive and scores poorly as the package outputted only neutral score across the 1,000 tweets that we have randomly sampled.

We compute the sentiment score per tweet as opposed to per sentence since given the limit of 140 characters per tweet, this is usually equivalent to one or at most two sentences. In cleaning up the raw tweets data, we remove special characters which include punctuation. *TextBlob* computes a sentiment score that ranges anywhere between -1 and 1. For our analysis, we bucket the scores to {-1, 0, 1} in order to produce stronger signals for feature engineering.

Keywords We enlist two sets of buy and sell keywords with approximately 30 words in each set. The keyword score is combined with the sentiment analyzer score to construct our signals.

- BUY SET = {’conviction’,’bold’,’up’, ’buy’, ’bullish’, ’bull’, ’free money’, ’long’, ’rise’, ’boom’, ’bid’, ’support’, ’grow’, ’get’, ’make’, ’earn’, ’investment’, ’invest’, ’investing’, ’invested’, ’buying’, ’bought’, ’pump’, ’like’, ’skyrocket’}

- SELL SET = {'scam', 'capitulation', 'down', 'fork', 'sell', 'short', 'bear', 'bearish', 'bubble', 'stop', 'crash', 'clamp', 'shut', 'freeze', 'fall', 'bust', 'trash', 'forbid', 'oppose','dash', 'sold', 'selling', 'collapse', 'plummet', 'plunge'}

Sentiment Score The following procedure applies to every single tweet in our data, which contains 1,000 tweets per day for the year 2018 in our main model. Each tweet contains the keyword "bitcoin" (case insensitive) and is emitted between 3 and 4pm of the day considered.

1. Compute keywords_score = *buy* keyword counts - *sell* keyword counts
2. Use sentiment analyzer to compute the "tone" of the tweet, which takes the value {-1, 0, 1}
3. Apply the following rules:
   - If tone is positive or neutral: return keywords_score as the tweet score
   - If tone is negative: return -(keywords_score) as the tweet score

	Positive Tone	Negative Tone
Keyword_BUY	Buy	Sell
Keyword_SELL	Sell	Buy

3.2 Model selection

Target From the above sentiment design and aggregation work on individual tweets, we derive hourly sentiment analysis features. As a reminder, due to the limitations of the *GetOldTweets* repository that we used to retrieve past tweets, we obtain this data only for one hour each day, between 3pm and 4pm - a serious follow-up to this project would be to stream tweets in real time using the official Twitter API to create this sentiment analysis data every hour and trade hourly. Therefore, natural targets for our sentiment-based trading strategy are the BTC-USD *hourly* forward returns. If there is signal in this "sentiment" data that we extracted from Twitter, this signal must take some time to be absorbed by the market (Coinbase in our case) and we expecte our features to be able to predict hourly BTC-USD returns a few hours forward to the last tweet retrieved. To illustrate this point, in our project we collect tweets between 3pm and 4pm every day and expect to predict the BTC-USD return between 4pm and a few hours later.

Signals The features we built correspond to *raw*, hourly, sentiment values. We want to transform them into sufficiently powerful and interpretable predictors of BTC-USD forward returns. The BTC-USD returns are variations in the BTC-USD asset price. Thus, rather than directly using the raw "buy" and "sell" sentiment signals, we subtract their moving averages over the past days[6]. This transformation has a mathematical interpretation: we are now looking at the discrete derivative of the "buy" and "sell" sentiment features. Intuitively, it means that *changes* in the social media sentiment about bitcoin can predict (to some extent) *changes* in the BTC-USD asset price (returns). This first transformation gets us two signals:

- **Buy Signal** the "buy" sentiment analysis feature with its moving average subtracted.
- **Sell Signal** the "sell" sentiment analysis feature with its moving average subtracted.

We then build a third signal which we call a **Normalized Sentiment Signal**, from the following formula:

$$\text{Normalized Sentiment Signal} = \frac{buy - sell}{buy + sell}$$

Where "buy" and "sell" are the *raw* sentiment features. The intuition behind this last signal is the following: the *balance* between positive and negative social media sentiment may matter more at predicting returns than each of the two Buy and Sell signals taken independently. Finally, following our intuition about variations we also consider this signal with its moving average subtracted.

[6]Here is another limitation from our work with 3 to 4pm data: we compute moving averages with *past days* sentiment values between 3 and 4pm. In a more comfortable scenario where we do have hourly tweets data, we would have computed moving averages over the previous hours, most likely with better results

Approach to alphas selection The next two steps are to evaluate and compare the predictive powers of our three potential alphas and to tune their hyper-parameters (window size for the rolling average, forward lag of the hourly returns to target in predictions). Rather than looking at classical metrics from supervised learning and statistics (such as the Mean Squared Error or the Coefficient of Determination R^2 coefficient of a signal taken as returns predictor) we turned to tools quite specific to algorithmic trading.

For each signal, we considered rolling average window sizes $s \in [3,\ 5,\ 7,\ 9]$ and drew the following 4 plots:

- **Scatter plot:** Simple scatter of the predictor against the return. Not specific to algorithmic trading, very noisy when many training points are available, but a standard starting point for investigations.
- **Binned plot**: refined version of the scatter plot, much less noisy and more informative. This plot is obtained by first organizing $(predictor,\ return)$ data points by "bins" or "buckets" of predictor values. In each bucket, the average of the forward returns is computed and plotted: we obtain a "binned" plot. By looking at the average return in each predictor bucket, we can evaluate to what extent this signal can predict returns. The observation of large return values in alpha buckets indicate a high predictive power of this alpha. One also hopes to notice a somewhat linear relationship between returns and the selected predictor to then incorporate this signal in a linear regression model for returns prediction.
- **Forward Biases plot**: the x-axis represents increasing lags for the forward returns, the y-axis values are the corresponding bias values. In our case, "lags" in the forward return correspond to hours forward. This plot tells us: for a given signal (based on tweets from 3 to 4pm), how many hours later does it best predict the return? The bias is the Pearson correlation coefficient between a given forward returns lag l and the predictor, times the standard deviation of the return:

$$bias(returns^{(l)}, pred) = \sigma_{returns^{(l)}} \times \rho_{returns^{(l)},pred}$$

- **Proxy Backtest plot**: cumulative sum of the quantity $returns \times predictor$ where the x-axis are time-steps in our dataset (daily timesteps for us). This plot is called a proxy backtest as it corresponds to the strategy of naively trading following our predictor, with no costs. One can show mathematically that the y value at the last time-step on such plot is equivalent to the bias point with the corresponding return lag on the forward biases plot. This plot also provides a qualitative look into the predictor's power: the "proxy PnL" points indicate how a strategy based on such a predictor would look like.

The plots obtained for our three signals and rolling window values $s \in [3,\ 5,\ 7,\ 9]$ are left in the Appendix. On those plots, you may note that the curves for the Sell Signal are "inverted" compared to the Buy Signal and the Normalized Signal. This is a good sanity check for the soundness of our predictors as, intuitively, "Sell" should be negatively correlated to the returns and the other two signal should be positively correlated to the returns, which is exactly what we observe. In the following paragraph, we present and discuss the plots obtained for the alphas retained in our project.

Alphas retained We first carried out our study of Sentiment Analysis predictors on data spanning the year 2018 only - *ie* approximately 365 data points. From the plots, all three indicators seemed reasonable to predict BTC-USD returns, although binned plots indicate a predictive power limited to a few percentiles of returns a few hours later. As the "Normalized Signal" presented the best forward biases plot and an encouraging linear relationship on the binned plot, we decided to pick this signal. We chose only one out of our three sentiment analysis predictors because they were by nature very correlated and putting several of them in a linear regression model would bring instability. We then looked into how volatility influenced the predictive power of our normalized signal: we computed the past 12 hours volatility for each data point, binned them into three buckets of "low", "medium" and "high" volatility, and computed again our evaluation metrics for the Normalized Signal *conditioned* on each of the three volatility buckets. The plots follow, and exhibit a behaviour that should have raised our awareness: the nice linear relationship between the Normalized Signal and BTC-USD returns is not that strong as it breaks down for "medium" volatility levels. Unsurprisingly, we also notice that for higher volatility levels, the magnitude of the predictive power of our alpha increases. We evaluated this alpha with a *rolling-forward* backtest with linear trading cost (assuming that our investments do not impact the market) and computed its Sharpe ratio on 2018 data. This evaluation procedure and those results are detailed in Section 5.

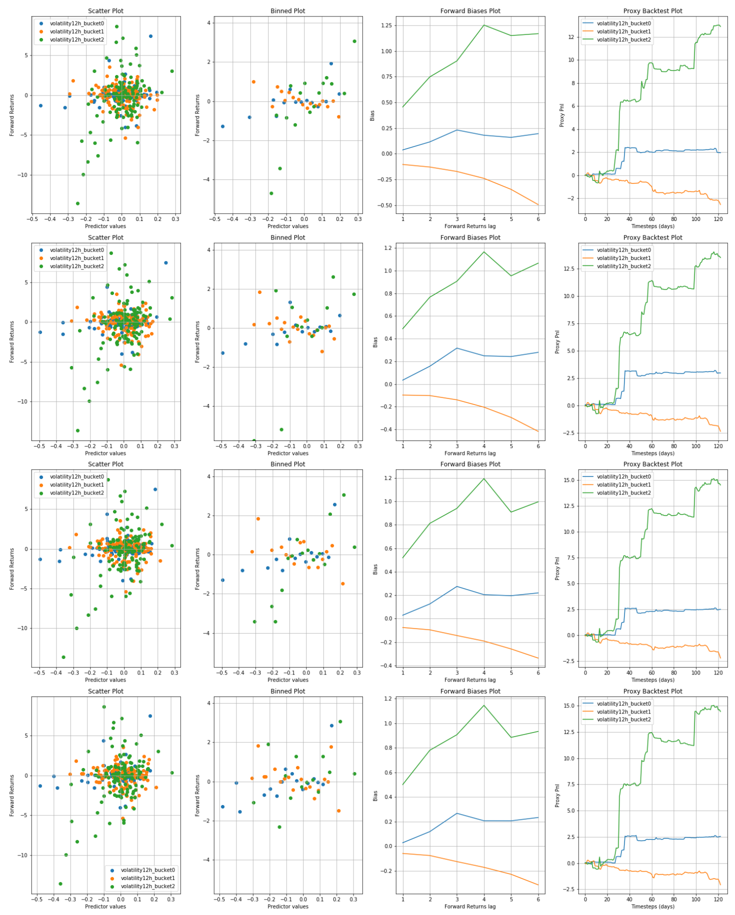


Figure 3: First alpha retained: Normalized Sentiment Signal conditioned on market volatility

After our final presentation, we went ahead and added 2019 data to our Universe. We quickly realized what the plots with conditioning on volatility suggested for the Normalized Signal: with the additional data points from 2019 the predictive power of this feature appeared weaker, a result confirmed by backtests on 2018 *and* 2019 data. The strength of the Buy Signal however, remained consistent after the addition of 2019 data, as the following plots show. Consequently, we picked it up as our final alpha for predicting forward BTC-USD returns and evaluated it on 2018 and 2019. Results are presented in Section 5, the Sharpe ratio of 0.761 with 5 bps of trading costs that we obtain is also more realistic. Results and backtest plots are examined in Section 5.

4 Work on Google Trends

4.1 Feature engineering

4.1.1 Data

To build our Google Trends signal, we query the search volume for 8 positive search keywords and 8 negative search keywords listed below, over the entire year of 2018, at the hourly granularity.

Most of those keywords were chosen as they were indicative of a positive or negative sentiment towards the cryptocurrencies market. We also included some exchanges such as "Coinbase" or "Bitmex", as we made the assumption that people need to connect to such platforms to trade

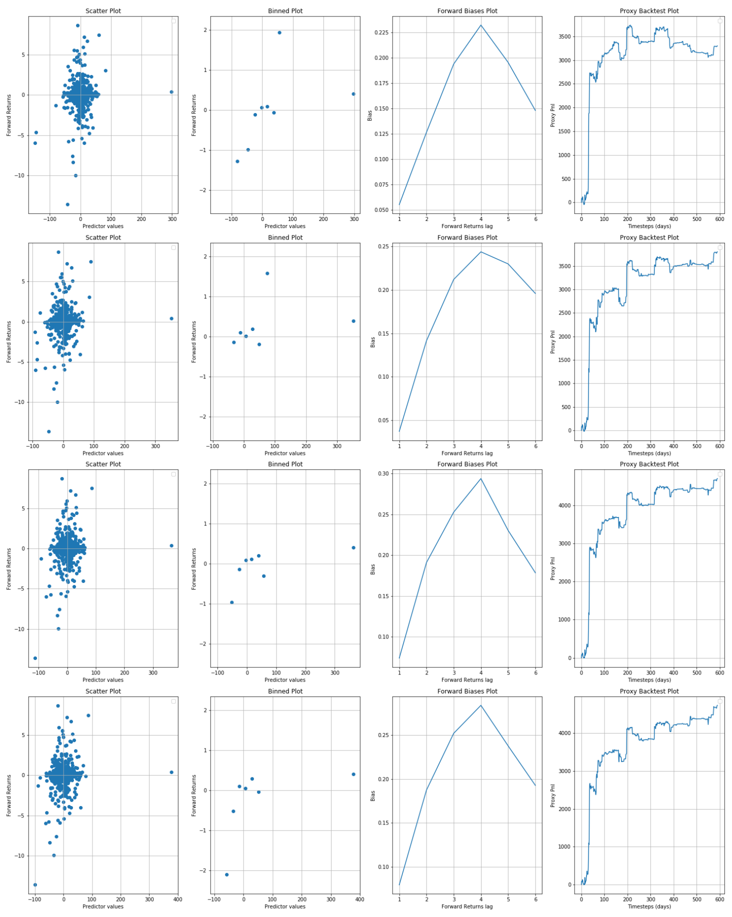


Figure 4: Second alpha retained: Buy Signal

Positive	Negative
"bitcoin price", "buy bitcoin", "bitcoin", "coinbase", "bitcoin boom", "make bitcoin", "bitmex", "long bitcoin"	"bitcoin crash", "bitcoin loophole", "sell bitcoin", "bitcoin scam", "short bitcoin", "bitcoin bubble", "bitcoin illegal", "bitcoin bear"

Table 1: Positive and negative keywords searches for Google Trends

cryptocurrencies.

4.1.2 Signal Construction

Once the previous data downloaded, we normalize each keyword search volume individually:

$$\text{Volume}_{\text{keyword}}(t) \leftarrow \frac{\text{Volume}_{\text{keyword}}(t)}{\max_t(\text{Volume}_{\text{keyword}}(t))}$$

We do this in order to put every keyword search volume to the same scale. Indeed, some keywords such as "bitcoin" naturally have a much larger volume than "Bitmex" for instance, but this does not indicate they would be more predictive of the returns.

After this step, we define the positive and negative volumes in the following way:

$$\text{Positive Volume}(t) = \sum_{\text{keyword} \in \text{Positive keywords}} \text{Volume}_{\text{keyword}}(t)$$

$$\text{Negative Volume}(t) = \sum_{\text{keyword} \in \text{Negative keywords}} \text{Volume}_{\text{keyword}}(t)$$

Finally, the signal is built by taking the normalized difference of the positive and negative search volumes:

$$S(t) = \frac{\text{Positive Volume}(t) - \text{Negative Volume}(t)}{\text{Positive Volume}(t) + \text{Negative Volume}(t)}$$

To ignore the overall market movement - cryptocurrencies were losing a lot of momentum in the beginning of 2018 - we substract the exponential moving average of the signal. We discuss in the next section how we adjust the half life of this moving average along with the horizon of this signal.

4.2 Model selection

4.2.1 Parameters tuning

To choose the horizon of our strategy, we look at the bias curve defined in the same way as in the previous Twitter section.

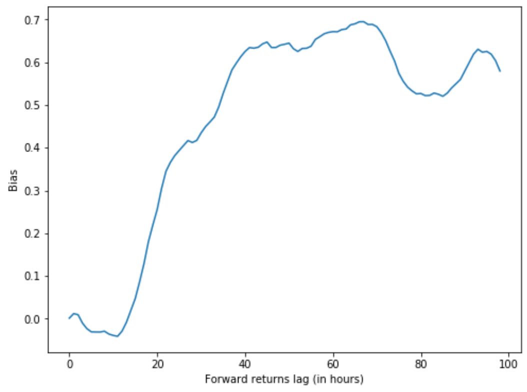


Figure 5: Bias as a function of the returns horizon

Based on the plot, it seems that our signal has the best predictive power at a horizon of 48 hours. We then try various values of the half life and settle with 24 hours. We only considered different scales of half life, such as 1 hour, 1 day, 3 days, and so on, as the difference between 24 hours and 25 hours for instance is negligible.

4.2.2 Results and comparison with Twitter signals

We now plot the signal against the forward return in 48 hours and see that the two seem to be positively correlated.
However, when we look at the proxy backtest we can see that the signal is very predictive only in the beginning of the year, but then flattens out and does not seem to be very predictive any longer.

This phenomenon can be explained by the much higher variance in the beginning of the year which leads to higher returns.

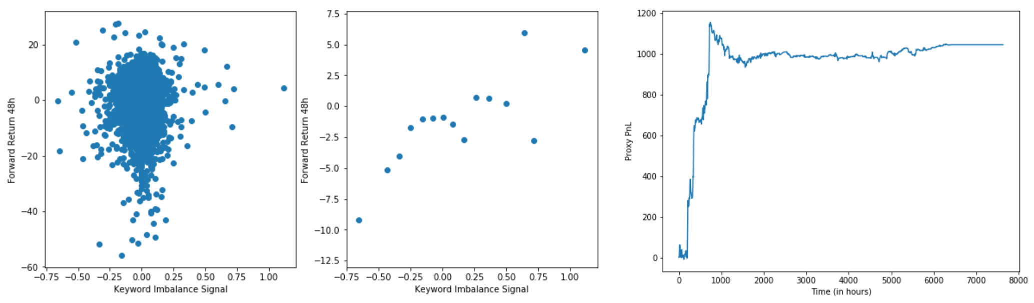


Figure 6: Scatter and Binned plot of 48h forward returns as a function of the signal, and proxy backtest through 2018

Another factor to take into account is that Google Trends captures an amount of hype/interest towards cryptocurrencies by the average trader who tends to search directly on Google if he should buy or sell bitcoins, instead of relying on more reliable analysis. While those kinds of users were very prominent in the beginning of 2018, just after the peak of December 2017 and when cryptocurrencies lost a lot of traction, we can assume they started to be less present in the second part of the year where the pool of traders made more informed decisions. On the other hand, those informed users tend to be more active on Twitter where they share more accurate analysis of the market, which is why the Twitter signal continues to be predictive for the rest of the year. For the next steps of this project, we decided to ignore this signal and focus only on the Twitter one.

To improve the Google Trends signal, we would need to conduct further research about the behavior of those more informed traders on Google to curate a list of keywords they may use and which would be more predictive of the returns.

5 Results

In the following section, we present the different models we built from the features derived after our work on Tweets data. We describe the trading strategy that we implemented and our evaluation model based on linear trading costs and a roll-forward backtest of two weeks window. We present backtest results and sharpe ratios of the different strategies obtained.

Models

On 2018 The linear model we have built on 2018 with our alpha research is

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3$$

where

- the target variable Y is the BTC-USD return 4 hours in the future predicted at 4pm
- $\beta_0 = 0$ as we want the the return to be null when we have no alpha
- X_1 is the normalized sentiment feature conditioned on the volatility regime of the past 12 hours being low
- X_1 is the normalized sentiment feature conditioned on the volatility regime of the past 12 hours being medium
- X_1 is the normalized sentiment feature conditioned on the volatility regime of the past 12 hours being high
- the predictors are computed using a moving average of 7 days

This model achieves an in-sample R^2 of 0.041

On 2018 and 2019 The linear model we have built on 2018 and 2019 is

$$Y = \beta_0 + \beta_1 X$$

where

- the target variable Y is the BTC-USD return 4 hours in the future predicted at 4pm
- $\beta_0 = 0$ as we want the the return to be null when we have no alpha
- X is the buy sentiment signal
- the predictor is computed using a moving average of 7 days

It turns out that the buy signal is overall more robust than the volatility-conditioned normalized signal, and we discuss this further below.

This model achieves an in-sample R^2 of 0.0285

Trading strategy The strategy used is a threshold-based strategy: at each timestep t, we either buy or sell the asset, given the value of the predictor at this time. We choose to trade a signed quantity q_t, constrained by a maximum exposure to the asset M, i.e. $|x_t + q_t| < M$ where x_t is the exposure at time t. We also suppose that the trading fees are linear, which is quite common when market impact and thereby slippage cost is negligible compared to fees or spread: when trading a signed quantity q_t, the fees are a linear cost $\Gamma|q_t|$ proportional to the traded quantity. The threshold is taken as the trading costs, hence yielding the following simple strategy:

- if $X_t > \Gamma$, we buy until reaching the maximum exposure M
- if $X_t < -\Gamma$, we sell until reaching the maximum exposure $-M$
- else, we do nothing

Backtest procedure We evaluate this model using a *rolling-forward backtest* with a two-week span. In other words, for each step $i \geq 0$ we train the model on the first $2i$ weeks, and evaluate it on the following two weeks $2i + 1$ and $2i + 2$. Hence, we adapt a cross-validation-like procedure to estimate our model's performance without ever looking ahead.

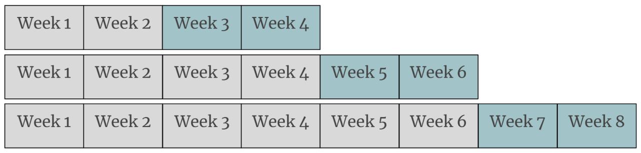


Figure 7: Rolling Forward backtest procedure

We run two backtest scenario, one with no trading fees and one with linear fees of 5 bps. From these backtests and its PnLs, we compute Sharpe ratios the following way:

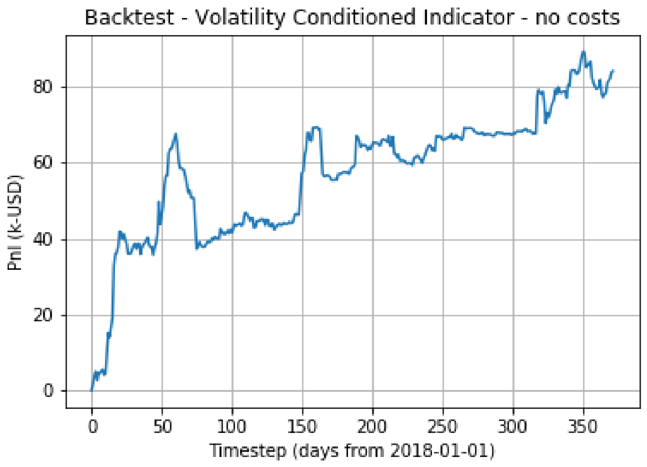
We construct a vector entry $PnL - cost \cdot turnover$ for every day. Given such a vector V where we concatenate all daily entries for an entire year, $\bar{V}/\sigma(V)$ yields the daily Sharpe ratio, that then get annualized by multiplying by $\sqrt{365}$ before being reported.

Results

On 2018 Here are the backtests and the Sharpe ratios we obtain from the model trained using 2018 data:

Sharpe Ratio	Trading costs (bps)
2.275	0.
1.708	5.

Table 2: Sharpe Ratios for the Normalized Indicator based strategy on 2018



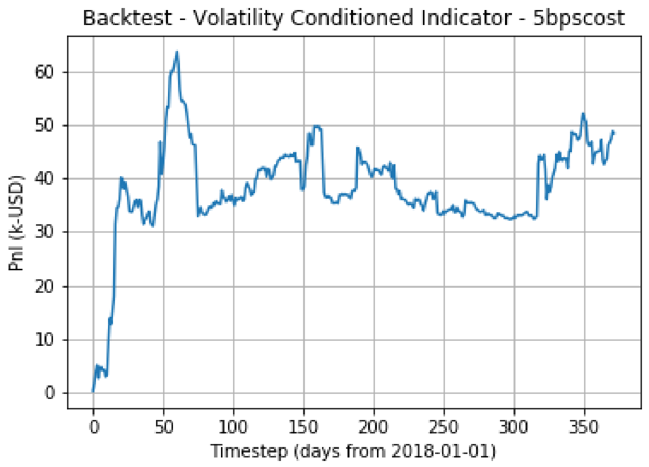


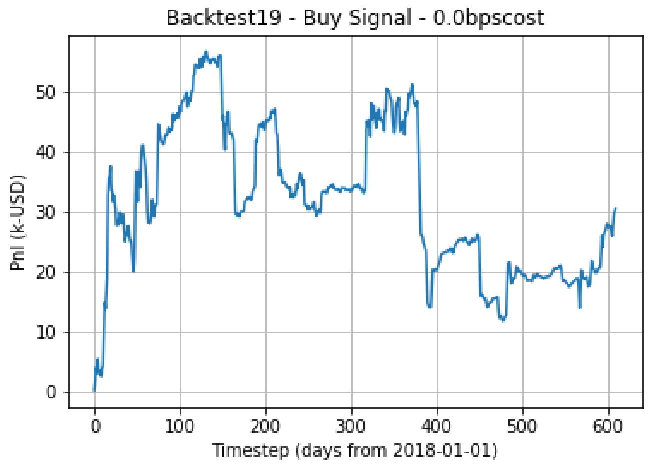
Figure 8: Rolling Forward Backtest of two weeks window over the year 2018, strategy on returns 4 hours forward, using the Normalized Signal (rolling average window of 7 days) conditioned on past volatility

We obtain high Sharpe ratio of greater than 1.7 with 5 bps trading fees. However, these results look a little too good to be true and realistic. In fact, our model built on 2018 is overfitting a dataset that is too small and representative of quite a peculiar market (indeed, early 2018 witnessed very high volatility). If we run this strategy on a backtest including 2018 but also 2019, we then actually lose money and obtain a Sharpe ratio of -0.087.

On 2018 and 2019 Given that our model built on 2018 was actually overfitting, we used more data and extended the dataset to include 2019. It turned out that the buy signal was overall the most robust alpha of the three we built, and that the conditionning on past volatility yielded good results by overfitting the market of 2018 which was quite exceptional, with record volatilities. This new more parsimonious model addresses this overfitting problem and yields the following backtests and Sharpe ratios:

Sharpe Ratio	Trading costs (bps)
0.567	0.
0.381	5.

Table 3: Sharpe Ratios for the Buys Signal based strategy on 2018-2019



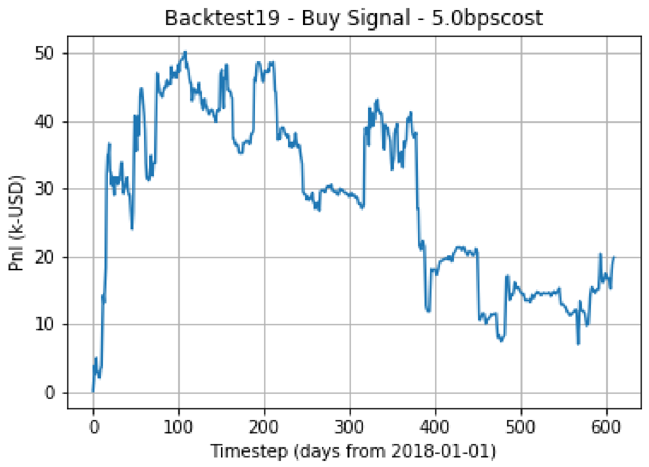


Figure 9: Rolling Forward Backtest of two weeks window from Jan 2018 to Apr 2019, strategy on returns 4 hours forward, using the Buy Signal (rolling average window of 7 days)

These backtests seem more realistic, with a Sharpe ratio of 0.381 supposing 5 bps trading fees. We expect this model to generalize better than the first one trained on 2018 only. In other terms, we think it will perform relatively well across more types of market behaviors.

In the future, we would be interested in continuing this project by further investigating this model. We would like to improve this strategy, without compromising its parsimony and overfitting to some market specifics.

Risk Management The price data on cryptocurrencies is fairly brief given the recent history of this asset class. This makes it challenging to accurately estimate correlation and Value-At-Risk (VaR) for the purpose of risk management. Based on the behaviour that has been exhibited in the period between 2017 and 2018, it is fair to ascertain that cryptocurrencies can be significantly correlated during market downturns, whilst less liquid alternative coins tend to outperform the major currencies during market upturn - perhaps as a result of the ICO bubble that propelled some token holders to vast wealth almost overnight. In terms of risk management, because our trading horizon focuses at the short term, the corresponding scaled hourly price volatility also contains less kurtosis and is thereby easier for the portfolio to monitor and respond to.

As part of our algorithmic portfolio construction, we set an alpha parameter threshold in determining when to enter and exit a trade. This parameter can be tuned to be more sensitive to price volatility such that through responsive portfolio turnover rate, the algorithm is able to reinforce its PnL position and risk control when counteracting a price shock. As an extension, we can also add a stop-loss parameter, as is common amongst fund managers, to automatically liquidate our positions once the loss reaches a certain percentage of our portfolio. Though this is an important feature to consider, given the limited data we have, more work needs to be undertaken with better data quality source (such as Level II market data from different crypto exchanges) in order to put in place accurate risk control limits. This is important for us especially if we are aiming to extend our trading horizon to multi-level longer term time scale, designing an accurate set of stop loss limits would be essential to the success of our fund.

6 Coupling Sentiment Analysis alphas with Market-based predictors

In order to improve the returns and risk metrics of our strategy, we decided to pursue on our research by looking at market data. We already pooled the data for backtesting purposes, so we decided to take advantage of that given the fact that a new data source tends to produce more orthogonal signals.

6.1 Long Term Horizon

To produce signals which are compatible with what we found on sentiment data, we started looking at the data's predictability on hourly or daily horizons. As a reminder the dataset that we have been working on include: opening price, closing price and volume of the cryptocurrencies with one min bins. In order to exploit this, we generated signals which translate into the following ideas:

- Momentum (i.e. trend following)
- Reversion
- Statistical arbitrage

Unfortunately, this didn't allow us to generate enough predictability over the long term. This is probably due to the crypto market being very speculative, there are very few "value factors" people consider to invest so we are not able to spot long term dynamics in the market data we had. However, these ideas proved accurate on a shorter time scale and still convinced us to explore a more high-frequency approach.

6.2 High-frequency Lead Lag

The term high-frequency is very ambiguous; in our case we refer to a subhour horizon, which would lead to a high strategy turnover with low return per trade (we trade multiple times a day but make small profit on each individual trade). Because of the law of large numbers, these strategies tend to have a much higher Sharpe ratio than low frequency strategies, but they are also more cost sensitive. We decided to focus on the lead lag effect of cryptocurrencies. We know crypto market is a special market, as the assets are very correlated but spread over a wide range of market caps. We would expect the event driven trading to occur on bigger cryptos first, which would justify why the prices of smaller cryptocurrencies would be driven by the larger ones.

We are interested in exploring the Bitcoin-Litecoin pair leadlag. We compute our lead lag signal, which is simply given by the past 5-min return on Bitcoin.

$$\alpha_t = \frac{p_{BTC}(t)}{p_{BTC}(t-5)} - 1 = r_{BTC}(t, -5)$$

We are going to use this to predict the 5-min forward return of litecoin, ie we model the following:

$$\mathbb{E}[r_{LTC}(t, 5)|\mathcal{F}_t] = \alpha_t$$

6.3 Results

The signal proved quite clean (easier to find a Sharpe signal on a short time scale). The relationship with the forward returns seems linear.

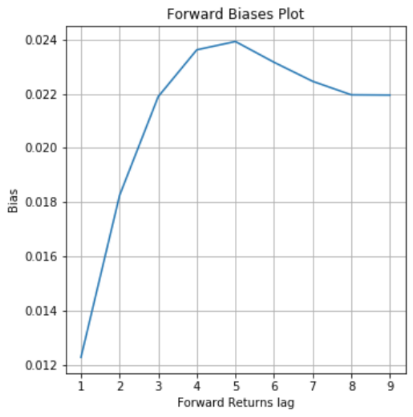


Figure 10: Bias as a function of horizon

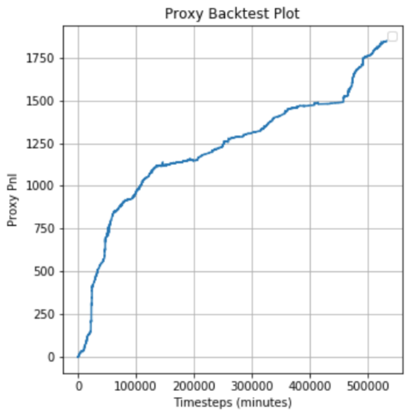


Figure 11: Proxy backtest in 2018

Looking at the forward biases plot, we see that our predictor seems to have a $\sim 5minhorizon.$

The proxy backtest seems consistent, and the high bias at this horizon will probably allow us to beat some trading costs. The proxy backtest seems better in higher volatility periods, but the high Sharpe is consistent over the entire dataset. We decided to backtest our strategy with 5 bps of linear costs.

As we can see, this strategy is still quite profitable, with a computed Sharpe Ratio of 38.5. It behaves as expected, and realizes such a profit because of its high turnover. The Litecoin is however very illiquid, and the strategy would not scale up. The actual turnover will be limited by liquidity, and a higher quantity will render the linear fees hypothesis unrealistic.

6.4 Extensions

Such a short-term strategy would clearly benefit from order book data. Some order book signals, combined with lead lag signals from other exchanges (especially the derivatives ones) could give us enough alpha to put together a full strategy. Another option could be to use this short term alpha as an indicator of how to execute the big orders sent by a more long term strategy.

7 Conclusion

Cryptocurrency trading was a very well suited topic for our exploration. With the low entry barriers, we managed to explore different data sources. The information we got from social and market data allowed us to set the grounds for realistic applications. As the market is growing, more hedge funds are becoming interested in trading cryptos, which share common properties with regular assets but still have their specificities (high volatility, high speculation, 24h trading, no centralization).

Our approach was very *feature driven*: we decided to opt for a simple model of the future returns (in our case, linear regression) but worked on specific features for them to reflect a financial intuition. This reduces the risk of overfitting, as we manage to comprehend our model and the financial grounds behind the trading strategy.

The sentiment based data gave interesting starting results, which could be worked on and improved. We do think though that this data can be more researched and looks promising. Refining our keywords, working with more data and different data sources (retweets, reddits, ...) could help us achieve a better predictability at the considered horizons. In this kind of mid-frequency strategy, we could also work on execution in order to pass a certain quantity to the market over a short period of time in a better way than simply sending one market order. This would enable us to get more scalability in our trading.

Over the short term, the market data exhibits some usual behaviour (in this case, lead lag effect). Threshold based execution is more adapted here, but there is still some work to do in terms of finding other predictors, and smartly choosing our trading threshold. The next step in this direction would be to look at microstructure dynamics to extract alpha. This data has to be pulled live from the exchange (we cannot get it historically) thus making it harder to obtain for research.

Acknowledgments

We are extremely thankful to the MS&E 448 teaching staff for guiding us through this project. Their assistance, their continuous feedback and support (pointing to potential pitfalls in our approach, encouraging our tedious initial endeavours with the data), and the resources they made available to us (courses materials, papers, the *GetOldTweets* repository) were incredibly valuable.

We would also like to thank all of the guest speakers who, coming from various backgrounds, shared with us their deep experience in quantitative finance and technology.

Ultimately, we are very grateful for an amazing and insightful class, all teams benefited from its stimulating and inspiring atmosphere to do an amazing job!

Appendix

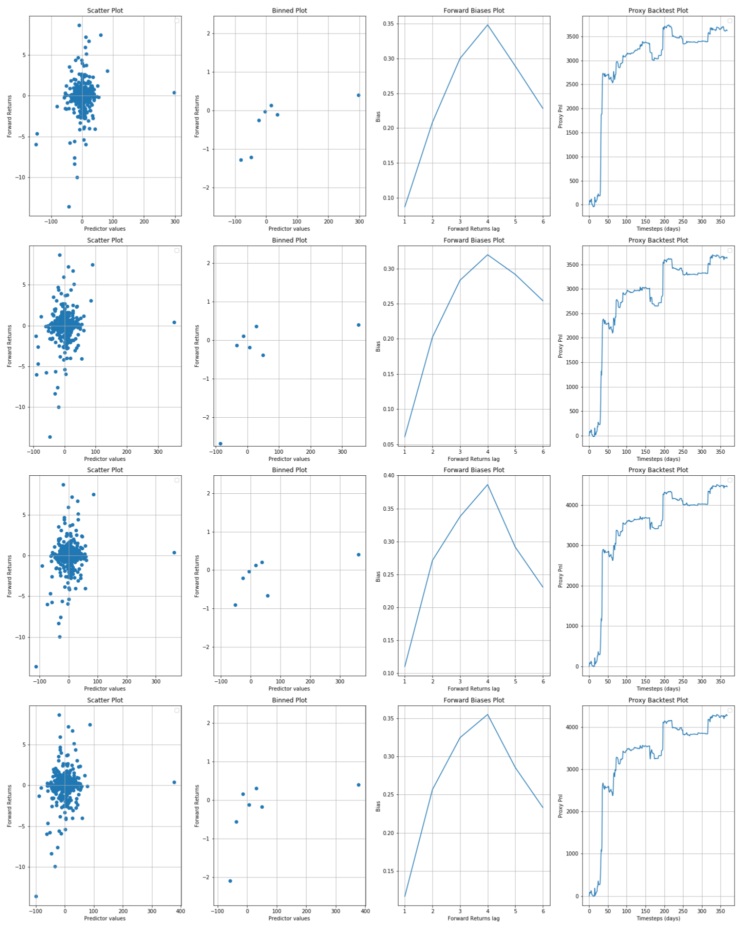


Figure 12: Buy Signal: Predictive power evaluation metrics

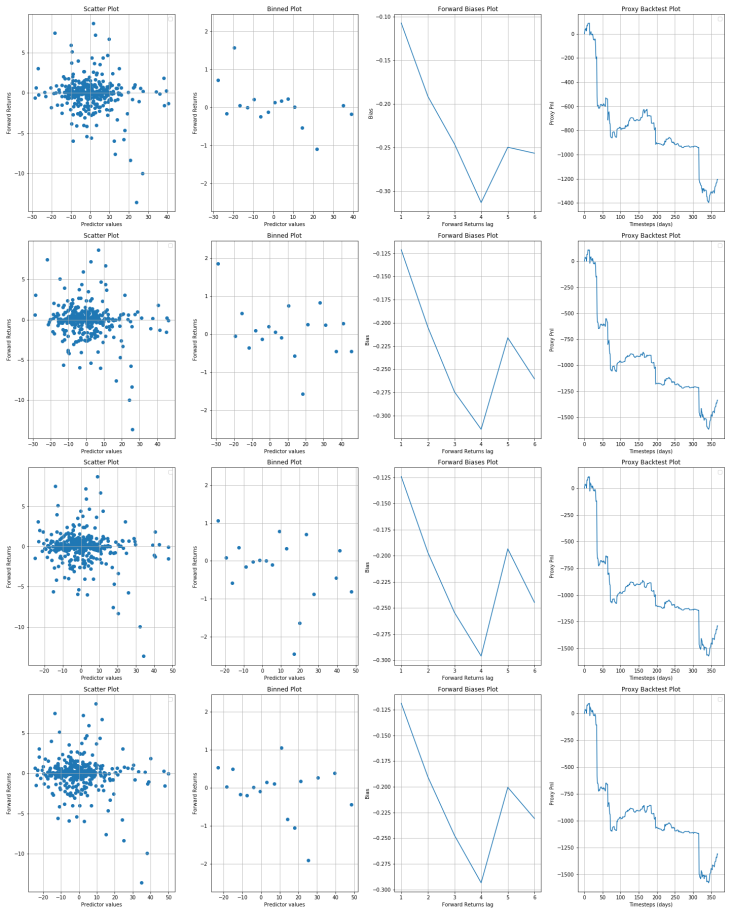


Figure 13: Sell Signal: Predictive power evaluation metrics

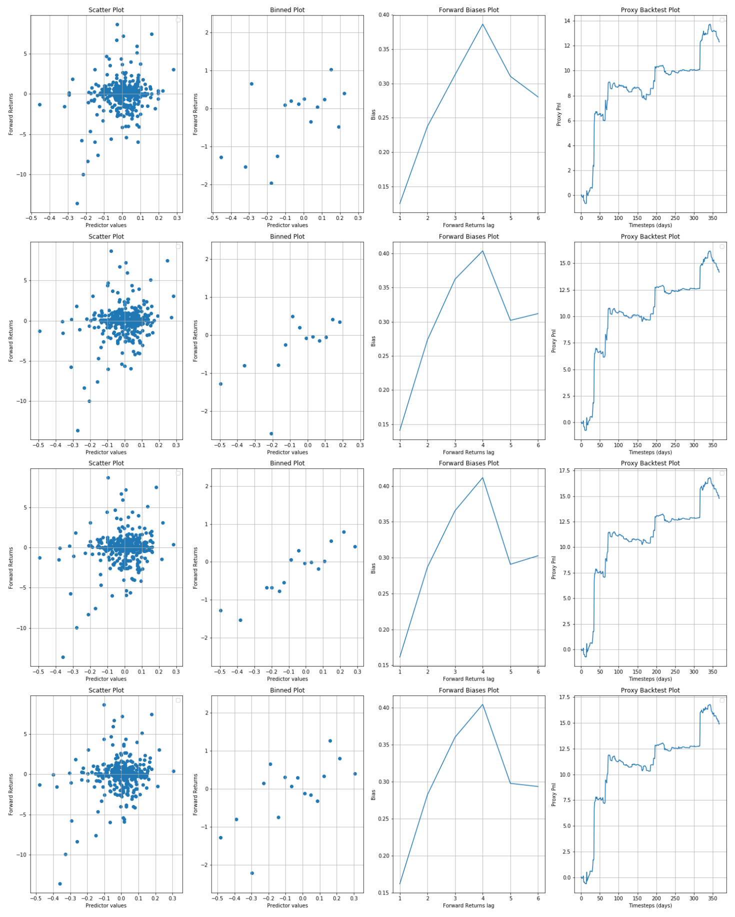


Figure 14: Normalized Signal: Predictive power evaluation metrics