Bitcoin sentiment analysis

Ambroise Thibault, Faune Blanchard, Maxime Lorenzo

Table of contents

1 Introduction

This project asks a simple question: can the mood of the market help us make better decisions when trading cryptocurrencies? We adapted a research paper called Ghost in the Machine Bybee [1], which showed that in traditional markets, sentiment often builds up before a crash. That made us wonder if we could take similar ideas and apply them to crypto, which is even more emotional and unpredictable. The goal was to see if tracking sentiment could help us understand how prices are going to move. Often, crypto naysayers say that cryptocurrency prices rely on nothing but speculation. Therefore, trying to understand the market's "irrational" response to news can help shed light on this debate. Even though it doesn't resolve the core question of the intrinsic value of bitcoin, it can show if investors in cryptocurrency are more or less emotional than investors in "classic" assets.

We started by collecting sentiment data from places like news headlines. The idea was to measure whether the overall tone on a given day was positive, negative, or somewhere in between. For this, we used the methodology presented in the paper, which we adapted to work with Mistral AI (the Large Language Model (LLM) developed by french startup Mistral), which is cheaper and easier to scale than GPT 3.5 (the LLM used in the paper). Feeding the headlines to the LLM helped us develop a time series of a sentiment "score" going from -1 to 1, which we can then compare to bictoin returns.

-> finish intro when we have final results

2 Generating sentiment score

2.1 Presentation of the database

We collected a dataset from GitHub entitled Crypto News Dataset. It aggregates daily headlines from a variety of crypto news websites. Each headline is labeled with the cryptocurrencies to which it refers. They are also timestamped down to the second the news was posted on the site. We also have the number of likes, dislikes, comments and others, although these are often 0, as well as the link to the article. In this project, we will only be using the raw headlines in order to generate the scores that we need.

Here is a quick view of the pre processed dataset with our function "import_data" in the "prompts" module. This function reads the csv files from the GitHub repository linked above, then it removes

unwanted columns and formats the date so that we only have the day at which it was posted. We can see that there can be multiple headlines in one day, or no headlines. This will cause issues that we will explain later on in the project.

The function also writes the prompts that we will later feed to the Large Language Model in order to properly generate the scores, which we will explain in the following section.

2.2 Generating prompts

For each headline, we generate a prompt which asks the AI its beliefs about the evolution of Bitcoin returns based on this headline. The prompt we decided to use is the exact prompt presented in Bybee [1] . The exact prompt reads :

```
Here is a piece of news:
"%s"
Do you think this news will increase or decrease BTC?
Write your answer as:
{increase/decrease/uncertain}:
{confidence (0-1)}:
{magnitude of increase/decrease (0-1)}:
{explanation (less than 25 words)}
```

Where "%s" is replaced by the headline. After editing this prompt, we notice that removing part of this prompt changes the results drastically (from "increase" to "decrease" for the same headline). Therefore, we decided to keep the prompt from the paper as is, even though we will only use the results for "increase" or "decrease" for this project. This prompt has been adapted from traditional retail investor surveys or surveys of CFOs that are used to produce a sentiment indicator. However, these are costly and slow. As presented in the paper, using AI models to replace human answers mimics the same results on a much bigger and faster scale.

More information about the construction on this prompt can be found in Bybee [1] .

Here, we use the Mistral LLM to generate prompt answers. The Mistral AI API is a service that gives us access to powerful language models, similar to ChatGPT or Claude. These models can read and interpret text, making them useful for tasks like summarizing news or detecting the tone of an article. In our project, we use the Mistral API to process financial headlines or crypto-related news and ask the model the prompt presented earlier. The model then gives us a structured answer, including its level of confidence and a short explanation.

Running sentiment scoring at a large scale using the Mistral API isn't just about sending a bunch of text to a model. It quickly becomes a technical challenge. If you're processing thousands (or even millions) of headlines, you need to manage API requests carefully. Language models are slow compared to normal scripts, and APIs often have strict rate limits. If you send too many requests too fast, you'll get throttled or even temporarily blocked. That means you need to build in delays, handle retries, and keep track of how many requests you're sending per second or minute.

To make this efficient, the code has to handle asynchronous execution and concurrency. Instead of sending requests one by one (which would make the code very inefficient and take a long time to generate answers), we wanted to send many in parallel—without breaking the API rules. That means using tools like Python's asyncio, aiohttp, or httpx, and writing logic that can pause, retry failed calls, and wait when needed. WWe however also needed to track which headlines have already been scored to handle failures cleanly. Scaling this up without crashing the system or getting banned by the API meant taking additional security measures in the code.

-> insert pseudo code of ambroise's final api pipeline

2.3 Scores methodology

After getting the scores from the LLM, we are left with categorical values for each news, while there are multiple news per day. Therefore, we decide to assign 1 if the LLM says the news will increase BTC, -1 if it says it will decrease, and if it is uncertain we will ignore the headline entirely (so as to not create

noise). Then, we aggregate every headline for one day, so that we get a clean series with one sentiment score per day. To aggregate, we use the formula presented in the paper:

$$F_t^{\text{gpt}} * (X_{t+h}^k) = \frac{\sum_{i \in \mathcal{A}_t} \mathbb{I}(\text{Increase}_i^k) - \mathbb{I}(\text{Decrease}_i^k)}{\sum_{i \in \mathcal{A}_t} \mathbb{I}(\text{Increase}_i^k) + \sum_{i \in \mathcal{A}_t} \mathbb{I}(\text{Decrease}_i^k)}$$

The EAR, or Ex-Ante Residual, is meant to capture the part of sentiment that can't be predicted based on past patterns. We do this by using a simple model (like an AR(1): a linear regression where the explanatory variable is simply the lagged target) to see what today's sentiment should be, based on yesterday's. Whatever's left over (i.e. the part the AR(1) model didn't expect) is the EAR. It's the surprise in the mood of the market.

This surprise is important because it often reflects emotional or irrational shifts. In crypto especially, people can get overly excited or scared in ways that don't line up with the data. By focusing on these unexpected jumps in sentiment, we're trying to isolate the moments when crowd psychology might be taking over—when prices could be getting ahead of reality.

3 Comparison with BTC returns

3.1 Extracting BTC returns

3.2 Comparing with scores

3.2.1 Initial results

3.2.2 Problems with our model

3.2.3 Interpretation

Conclusion

[1] J. Leland Bybee. *The Ghost in the Machine: Generating Beliefs with Large Language Models*. 2025.