OBJECTIVE

Develop a bitcoin trading strategy that is significantly more accurate than a random guess and therefore the excess return is not attributed to chance. The corollary is the strategy should generate alpha with respect to a HODL strategy. Bitcoin price is highly volatile; a trading strategy that can consistently limit downside deviation could generate outsized returns. Daily returns are tough to predict with accuracy, so I applied some unconventional techniques while taking care to properly handle time series data.

DEFINING THE STRATEGY

Build machine learning processes to predict whether tomorrow’s price (t + 1) will be positive or negative. The strategy is simple:

* ONE security: bitcoin
* ONE binary signal: 0 = short, +1 = long
* ONE-step-ahead: forecast only tomorrow’s return

The underlying assumption is that the full position is either bought/held or sold at the close of each day based on trade signal.

FEATURE ENGINEERING

An important aspect of time series analysis is feature engineering. My features can be grouped into three categories – lags, technical indicators, and cointegrated assets. There are a few that fell outside of these categories, but they turned out to be relatively unimportant (I will get to this later).

**Lags.** The acf and pacf plot on log daily returns showed significant lags at 6 and 10 at the 95% confidence level, however, there was no clear AR or MA pattern, so I made a grid search function and selected the order with the lowest AIC score, ARMA(8,5).

**Technicals.** I included 7, 14, 20, and 50-day SMA and RSI features.

**Cointegrated assets.** Even if two prices follow a random walk, it's possible that a linear combination of them does not follow a random walk. So even though the individual prices themselves are not forecastable because they're random walks, the linear combination is forecastable. I tested for cointegration of various asset prices with bitcoin prices.

I did this in two parts: First, I regressed bitcoin prices on other asset prices to get the slope coefficient m. Second, I ran the Augmented Dickey Fuller test on the linear combination of each series pair to determine whether it's a random walk. The null hypothesis is no cointegration. I set a 10% significance level for rejecting the null. I included assets that passed the test in the chart below. The top chart is log price of the assets and the bottom is the spread (bitcoin price – m\*asset price), mean subtracted and centered at zero. Each shows clear mean reversion. This implies that there is an economic relationship between these assets and bitcoin such that including them in the model should improve the forecast. Last, looking at Chinese equities and emerging market equities, the spread to bitcoin is almost exact. Because Chinese equities and emerging market equities are strongly correlated and because China accounts for 2/3rd of bitcoin mining alone, I used Chinese equities only (dropping EM equities).

|  |  |
| --- | --- |
| **Cointegration Plots** | |
| A screenshot of a cell phone  Description automatically generated | A screenshot of a cell phone  Description automatically generated |
|  |  |
| A screenshot of a cell phone  Description automatically generated | A screenshot of a map  Description automatically generated |

MODEL DEVELOPMENT

Linear Models

Non-linear Models

RESULTS ANALYSIS

Signal vs Benchmark – tracking buy and sell decisions, cumulative, and rolling return

|  |
| --- |
| A close up of a map  Description automatically generated |
|  |
| A close up of a map  Description automatically generated |

|  |
| --- |
| A screenshot of a cell phone  Description automatically generated |
|  |
| A screenshot of a cell phone  Description automatically generated |

Randomness Simulation – model accuracy vs random chance

|  |
| --- |
| A screenshot of a cell phone  Description automatically generated |