

1. Introduction
   1. Context and motivation

I first entered the crypto market in late 2020, driven by a fascination with this nascent asset class and its corresponding opportunities, bringing a new incremental technology that Blokchains represent. Unlike established markets, cryptoassets present a unique combination of innovation, unpredictability and volatility, reflecting the characteristics of an emerging market. These include high volatility, rapid technological evolution, and an ecosystem often influenced by sentiment-driven behavior, and the other way around.

One of the most striking observations I made early on was the cyclical nature of Bitcoin’s price movements. While historical cycles exhibit similarities, such as periods of extreme euphoria followed by sharp corrections, they also show distinct characteristics as the market evolves. Within this apparent complexity lies a fascinating simplicity: a form of seasonality where fear and greed seem to drive prices to extremes. From this, a compelling hypothesis emerged—longing during extreme fear and shorting during extreme greed could serve as a highly effective trading strategy.

[A graph on a black background

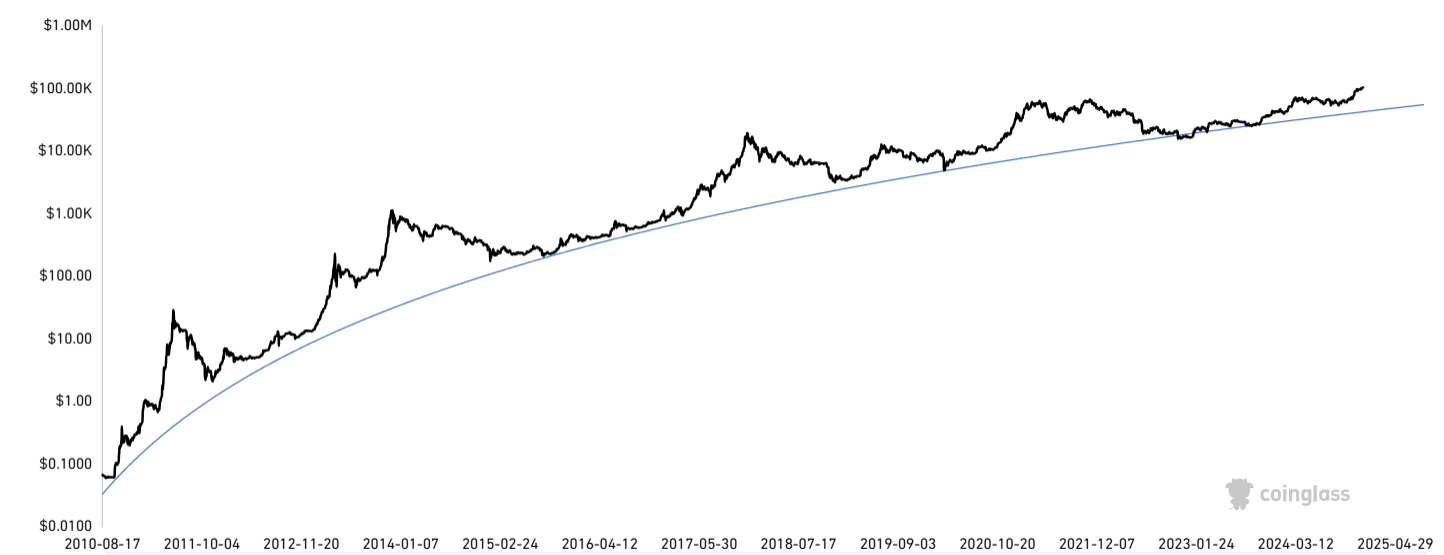
Description automatically generated](https://charts.bitbo.io/fear-greed/)

As I delved deeper, it became clear that the cryptomarket is rife with potential weak-form inefficiencies, a hallmark of an emerging market where participants are often swayed by emotion rather than rational analysis. These inefficiencies create fertile ground for exploring technical analysis and behavioral patterns that might not hold as strongly in more mature financial ecosystems. Motivated by this insight, I set out to design an indicator that captures these emotional extremes and reliably predicts Bitcoin’s tops and bottoms.

Before continuing, I highly recommend for newcomers this [2024 cryptomarket introduction guide](https://www.linkedin.com/posts/benjamin-faccin_cryptoinvesting-web3-investmentguide-activity-7242920032156954626-4yC-?utm_source=share&utm_medium=member_desktop) I’ve published and my following report for curious: [*Era of interoperability: no matter what, but when?*](https://www.linkedin.com/posts/benjamin-faccin_era-of-interoperability-no-matter-what-activity-7116424236675543040-cvgr?utm_source=share&utm_medium=member_desktop) (2023).

* 1. Concerns and objectives

The main concern isn’t about accurary of the current model, it’s about how long this model will be accurate enough. As evoked earlier, historical Bitcoin patterns’ cycles are very similar in terms duration and very different at the same time, especially in terms of expansion which tends to be logarithmic. This affects a lot the model as some independent variables I’ll use will show less extreme values through time.

[](https://www.coinglass.com/fr/pro/i/bitcoin-rainbow-chart)

While the concept of capitalizing on extremes of fear and greed is intuitive, translating this into a systematic and actionable tool requires careful analysis and validation.

Additionally, the cryptomarket’s relatively short history poses a significant constraint, as the limited dataset may lead to overfitting or misinterpretation of patterns. Therefore, another key concern is ensuring that the indicator accounts for false signals, which could erode its effectiveness over time.

That is why I will update the model after every new local bottom\* and local top\*. This will enhance the ability to capture updated new trends, coefficients, seasonalities and other patterns up-to-date.

The main objectives of this model will be to have both performance and validation very good metrics. Threshold, criteria and t-tests used are developed in the third chapter.

**Disclaimer**

This research and the indicator developed are intended for informational and educational purposes only and should not be considered financial advice. Trade at your own risk.

\*A new local top will be set if the price goes below the last cycle all-time-high (I will take the local ATH from the current cycle).

\*A new local bottom will be set if the price goes above the last ATH (I will take the local ATL from the current cycle).

1. Indicators
   1. List of independent variables used

The selection of independent variables was painstaking and reworked progressively during testing. The initial, non-exhaustive list included around twenty technical indicators, but the final version contains exactly seven. All the indicators used come from TradingView, with most being community-created indicators.

**Net Unrealized Profit and Loss (NUPL)**

* **Description:** Net Unrealized Profit and Loss (NUPL) is a financial metric used to assess the overall profitability of holders of an asset by analyzing unrealized gains and losses. It calculates the difference between unrealized profits (when the asset's current price is higher than the purchase price) and unrealized losses (when the purchase price exceeds the current price), relative to the market capitalization. A positive NUPL indicates that the majority of holders are in profit, while a negative NUPL suggests most are at a loss. This metric is widely used to gauge market sentiment and identify potential trend reversals. High or low extremes in NUPL often signal overbought or oversold market conditions, respectively.
* **Credits:** [VanHe1sing](https://www.tradingview.com/u/VanHe1sing/)

**Spent Output Profit Ratio (SOPR)**

* **Description:** Spent Output Profit Ratio (SOPR) is a financial metric that measures the profitability of coins moved on a blockchain. It is calculated as the ratio of the selling price (realized value) to the purchase price (cost basis) of spent outputs. A SOPR value above 1 indicates that holders are selling at a profit, while a value below 1 suggests sales are at a loss. This metric helps identify market sentiment, with high SOPR values signaling profit-taking phases and low values indicating capitulation or undervaluation. SOPR trends are often used to predict potential market reversals or confirm ongoing trends.
* **Credits:** [Pinnacle Investor](https://www.tradingview.com/u/Pinnacle_Investor/)

**Market Value to Realized Value Z-score (MVRV-Z)**

* **Description:** The MVRV-Z Score is a blockchain metric used to assess whether an asset is overvalued or undervalued relative to its "fair value." It is calculated by comparing the Market Value (market capitalization) and Realized Value (the aggregate value based on purchase prices), with a Z-score standardization to highlight deviations. A high Z-score indicates overvaluation, often signaling market tops, while a low Z-score suggests undervaluation, potentially indicating market bottoms. This metric is particularly useful for spotting extremes in market sentiment and identifying potential buying or selling opportunities. It is widely used in cryptocurrency analysis for long-term trend evaluation.
* **Credits:** [Da Prof](https://www.tradingview.com/u/Da_Prof/)

**Cumulative Coin Value Days Destroyed (CVDD)**

* **Description:** Coin Value Days Destroyed (CVDD) is a blockchain metric that measures the economic activity of spent coins by combining their value with the time they were held. It is calculated by multiplying the number of coins in a transaction by the number of days since they were last moved, effectively capturing the "weight" of long-term holders selling. Higher CVDD values suggest significant activity from long-term holders, which may indicate profit-taking or changes in market dynamics. This metric is often used to analyze holder behavior, assess market sentiment, and identify potential turning points in the market. It provides insights into the maturity and movement of assets over time.

Following indicator rebases the Cumulative Coin Value Days Destroyed with a shift bottom extension equals to 120% of original CVDD and a top extension approximatevely equals to 352% of original CVDD. Also, following CVDD was calculated with a denominator of 2.2\*10e7, which is uncommon: either 6.0\*10e6 or circulating supply are commonly used.

* **Credits:** [Da Prof](https://www.tradingview.com/u/Da_Prof/)

**Simple Moving Averages**

* **Description:** The SMA20, SMA50, SMA100, and SMA200 are Simple Moving Averages calculated over 20, 50, 100, and 200 periods, respectively, and are widely used in technical analysis. These averages smooth out price data to identify trends and potential support or resistance levels. The SMA20 reflects short-term trends, the SMA50 highlights intermediate trends, the SMA100 indicates medium-term trends, and the SMA200 focuses on long-term trends. When shorter SMAs cross above longer ones (e.g., SMA20 crossing SMA200), it signals potential bullish momentum, while the reverse suggests bearish trends. Combining these SMAs provides a comprehensive view of market trends across different time horizons, aiding in better-informed trading decisions.
* **Credits:** TradingView own indicator

**Relative Strength Index (RSI-14EMA)**

* **Description:** RSI 14EMA (Relative Strength Index with a 14-period Exponential Moving Average) is a momentum oscillator used to measure the speed and magnitude of price changes. It calculates the ratio of recent gains to losses over 14 periods and applies an Exponential Moving Average for smoother, more responsive signals. RSI values range from 0 to 100, with readings above 70 indicating overbought conditions and below 30 suggesting oversold conditions. The EMA adjustment makes it more sensitive to recent price movements, enhancing its utility for short-term analysis. RSI 14EMA is widely used to identify trend strength, potential reversals, and entry or exit points in trading.
* **Credits:** TradingView own indicator

**Bitcoin Difficulty Ratio (14EMA)**

* **Description:** Bitcoin Difficulty Ratio (14EMA) is a metric that tracks the relationship between Bitcoin's current mining difficulty and its historical trends, smoothed using a 14-period Exponential Moving Average. It highlights changes in the computational effort required to mine blocks, reflecting network security and miner activity. A rising Difficulty Ratio often signals growing network strength and optimism, while a declining ratio may indicate reduced miner confidence or profitability. The 14EMA makes the metric more responsive to recent changes, aiding in short-term analysis. This ratio is commonly used to assess market health and potential shifts in miner sentiment or network dynamics.
* **Credits:** [gliderfund](https://www.tradingview.com/u/gliderfund/)

To build the model, I had to export all historical data on a daily and weekly basis from the first recorded value for each indicator. However, this raises two issues regarding the long-term sustainability of my model:

* How can I update future data if the owner of an indicator decides to make it no longer publicly available or deletes it?
* How can I update future data if the API used by the indicator's owner, which is often subscription-based, becomes obsolete or if their subscription is canceled?

For these reasons, after identifying all the required indicators that are both independent and essential for my model's performance, I decided to replicate them one by one using Python.

* 1. Python Replication

Exportation from both APIs ([bitcoin-data.com](https://bitcoin-data.com/v1/difficulty-BTC)) and complemented with fallback data from [Blockchain.info](https://blockchain.info/charts/difficulty) are on a daily basis, and I computed daily data to get weekly ones. Weekly data for all indicators was derived by identifying Mondays within the dataset. When data for a specific Monday was unavailable, the closest preceding value within a seven-day range was used. For all indicators, missing values were forward-filled, ensuring continuity and reliability across the entire time series.

Several factors can influence the replication process and contribute to discrepancies between TradingView values and replicated values. Although not exhaustive, one common issue arises when an API fails to provide data for a specific week. This can disrupt calculations such as moving averages or shift the reference point for the end of the week. Additionally, the chosen trading pair can introduce variations; APIs typically use the BTC/USD pair from a specific exchange, while TradingView often employs the aggregated “BTC/USD Crypto” pair, which combines data. Last but not least, there are differences in computation methods by APIs, such as distinct formulas or division factors (e.g., for the CVDD indicator), that increase delta.

However, overall, results are pretty satisfying. A huge thanks to bitcoin-data, which is the only one I’ve found that provides free various data (30 requests limit per hour).

**Net Unrealized Profit and Loss (NUPL)**

* **Calculation**: A percentage that measures the overall profitability of Bitcoin holders in the market.

A black text on a white background

Description automatically generated

A graph of a stock market

Description automatically generated

**Spent Output Profit Ratio (SOPR)**

* A black and white image of a symbol

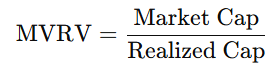
  Description automatically generated**Calculation**: A smoothing mechanism was applied using the exponential moving average to compute Signal Line:

A graph with blue and red lines

Description automatically generated

**Market Value to Realized Value Z-score (MVRV-Z)**

* **Calculation**: The Z-score normalization of MVRV (historical standard deviation).

A math equations on a white background

Description automatically generated

A graph of a graph

Description automatically generated

**Cumulative Coin Value Days Destroyed (CVDD)**

* A black text on a white background

  Description automatically generated**Calculation**: CVDD incorporates cumulative value days destroyed (CDD) and market age. Adjusted CVDD was obtained by scaling CVDD (150%).

A black and white image of a mathematical equation

Description automatically generated

A graph of a graph

Description automatically generated

**Simple Moving Averages**

* A math equation with numbers and symbols

  Description automatically generated**Calculation**: Simple moving averages (SMA) were computed over different windows (20, 50, 100, 200 days).

A graph of a stock market

Description automatically generatedA graph of a stock market

Description automatically generated

**Relative Strength Index (RSI-14EMA)**

* A number and line with black text

  Description automatically generated with medium confidence**Calculation**: Gains and losses were computed using exponential weighted averages over a 14-day period:

A black and white text

Description automatically generated

A graph with blue lines

Description automatically generated

**Bitcoin Difficulty Ratio (14EMA)**

* **Calculation**: A smoothing mechanism was applied using the exponential moving average:

A math equation with numbers and symbols

Description automatically generated

A graph with a line

Description automatically generated

A graph with a red line

Description automatically generated

* 1. Normalization of the dependent variable

The dependent variable must be normalized to be viable for use and to train the neural network model. In this case, after experimenting with several types of normalization that were not sufficiently precise, I decided to normalize my independent variable on a scale ranging from -1 to 1, where -1 corresponds to the lowest level of a cycle and 1 corresponds to the highest level of a cycle. I used linear interpolation between each extreme to assign intra-top-bottom variables. The first data point begins in early 2014, and the last corresponds to the lowest level of the most recent cycle: November 2022.

Visually, the normalization of the dependent variable relative to the price appears as follows:

A graph of a graph

Description automatically generated with medium confidence

What you can immediately see is the non-consideration of second 2021-top, as on-chain metrics regarding my independent variables were too much contradictory.

Now that the dependent variable and independent variables are set, successfully replicated and automatically updated, let’s introduce *Eclipse.*

1. Methodology
   1. Long Short Term Memory neuronal model

Long Short Term Memory (LSTM) is a type of recurrent neural network (RNN) specifically designed to learn and retain patterns in sequences of data over long time periods. Traditional RNNs struggle with the "vanishing gradient" problem, which limits their ability to learn dependencies in long sequences. LSTMs solve this by using a unique architecture with memory cells that can store, update, or forget information over time. These cells are controlled by three gates: input, forget, and output gates, which regulate the flow of information into, out of, and within the cell, respectively.

**How LSTMs Work:**

1. **Forget Gate:** Decides which information in the memory cell should be discarded by analyzing the current input and previous hidden state.
2. **Input Gate:** Determines what new information should be added to the memory cell. This involves a candidate value (a potential memory update) and a modulation of its influence.
3. **Output Gate:** Controls what part of the stored memory is used to compute the current output and the next hidden state.  
   By carefully combining these gates and updating the cell state, LSTMs can maintain relevant information while discarding irrelevant details, enabling them to capture long-term dependencies in sequential data.

LSTMs are particularly well-suited for modeling and predicting the cyclical tops and bottoms in Bitcoin markets, forming the foundation of the "Eclipse" indicator. Bitcoin price movements are characterized by temporal dependencies and recurrent cycles influenced by macroeconomic factors, market sentiment, and blockchain-specific events, such as halvings. The Eclipse indicator leverages the LSTM's architecture to analyze and interpret these time-dependent dynamics, capturing both short-term fluctuations and long-term trends in the data.

The Eclipse indicator capitalizes on the LSTM's ability to process sequential data and recognize nonlinear relationships between given variables. By selectively retaining and discarding information through its gating mechanisms, the LSTM can identify key patterns and turning points in Bitcoin’s price cycles. This allows the Eclipse model to provide robust predictions of potential market reversals, offering valuable insights for researchers and market participants seeking to anticipate cycle tops and bottoms with precision.

* 1. Data and pretreatment

The data utilized comprises all previously mentioned and replicated datasets. These are updated on a weekly basis, with no additional preprocessing applied, apart from the rebasing to 100 on row data, as outlined in the section detailing the replication of certain variables. It was then applied a MinMaxScaler with a range of (0, 1) as a normalization used to scale the values of independent variables to lie within the interval [0, 1]. It is a preprocessing step commonly applied before training machine learning or deep learning models to improve convergence and model performance.

MinMaxScaler used transforms each independent variables values using the formula:

A mathematical equation with black text

Description automatically generated

I developed a Python script that consolidates all tasks required to update, import, and save files for each independent variable. For reporting purposes, the script is executed every Wednesday, ensuring that data from the preceding Monday is available, thereby operating on a D+2 basis.

* 1. Structure and parameterization

The "Eclipse" LSTM model is designed to predict cyclical patterns in Bitcoin markets using time-series data. The model architecture and its parameters have been carefully configured to ensure robustness and reliability in capturing temporal dependencies and nonlinear relationships. Below is a detailed explanation of its structure and parameterization:

**Input Data**

In the development of the Eclipse indicator, a 52-week sliding window was selected as the optimal timeframe for modeling. This choice allows the model to capture a full year of weekly data, effectively incorporating seasonal trends and annual cycles that are critical in Bitcoin's market behavior. The 52-week window strikes an ideal balance between preserving sufficient historical context to identify long-term patterns and maintaining the responsiveness needed to reflect recent market dynamics.

Extensive testing was conducted with various window lengths, ranging from shorter to longer periods. These experiments revealed that shorter windows lacked the necessary context for detecting longer-term cycles, while longer windows introduced excessive noise, diminishing the model's ability to adapt to recent data. The 52-week window consistently produced the best performance and accuracy, making it the most effective choice for predicting cyclical tops and bottoms in Bitcoin’s market trends.

A screenshot of a chat

Description automatically generated

**Model Architecture**

The use of 128 units provides a robust capacity for learning complex sequential patterns and relationships in the data. A higher number of units allows the model to capture subtle long-term dependencies and ensures that sufficient representational power is available for the feature-rich input.

Setting return\_sequences=True is necessary to pass the entire sequence of hidden states to the next LSTM layer. This facilitates stacking multiple LSTM layers, enabling the model to learn hierarchical temporal features and improving its ability to detect multi-scale patterns in the data

This shape reflects the model's input: 52 time steps (representing one year of weekly data) and 7 features (the selected independent variables). This configuration allows the model to effectively analyze relationships between features across the entire sliding window.

A reduced number of units (64 compared to the first layer's 128) helps to distill and compress the information learned in the previous layer. This reduction minimizes overfitting while retaining critical temporal dependencies, enabling the model to focus on the most relevant patterns for prediction.

With return\_sequences=False, this layer outputs only the final hidden state rather than the full sequence. This is appropriate for downstream dense layers that require a fixed-size input, such as the output layer used in this model.

The dense layer consists of a single neuron, which produces the final prediction for the dependent variable. This ensures a one-to-one correspondence with the target output.

A screenshot of a computer

Description automatically generated

**Training Configuration**

A batch size of 32 balances computational efficiency and model performance, with a maximum of 100 epochs. However, Early Stopping halts training if the validation loss does not improve for 10 consecutive epochs, restoring the best weights to prevent overfitting. The data is split into 80% training and 20% validation, ensuring the model is evaluated effectively while being trained on a substantial portion of the data.

A screenshot of a computer

Description automatically generated

1. Experimental results
   1. Performances analysis

The performance metrics indicate that the Eclipse model achieves exceptional predictive accuracy and robust generalization capabilities. On the training set, the MAPE of 5.26% reflects that, on average, the model’s predictions deviate by 5.26% from the true values. This implies that for a target value normalized between -1 and 1, the average absolute deviation remains small, typically within the range of ±0.0526 in scaled terms. Similarly, the R² value of 0.9989 signifies that 99.89% of the variance in the dependent variable is explained by the model, demonstrating a near-perfect fit. The RMSE (0.0189) quantifies the average magnitude of prediction errors, with its low value indicating minimal large errors, while the MAE (0.0143) shows that, on average, the model’s absolute prediction error remains close to 0.0143 in scaled terms.

For the validation set, the MAPE of 5.81% suggests a slightly higher average percentage error on unseen data, which translates to a mean absolute deviation of approximately ±0.0581 within the normalized range. This still reflects strong predictive capability. The R² value of 0.9988 indicates that 99.88% of the variance in the validation data is captured, confirming the model's ability to generalize effectively. The RMSE (0.0208) suggests that the model’s average prediction errors are slightly larger on unseen data, but still negligible in magnitude, and the MAE (0.0163) confirms consistently low absolute prediction errors.

These results imply that the model reliably predicts cyclical tops and bottoms within a small margin of error, even when applied to new data. The low MAPE ensures that deviations remain proportionally small, the high R² demonstrates the strength of the model in explaining variability, and the low RMSE and MAE indicate that predictions are precise and consistent across both training and validation sets. This makes the Eclipse model a scientifically robust and practically reliable tool for forecasting market cycles.

A screenshot of a computer

Description automatically generated

* 1. Assumptions analysis

The statistical tests performed on the residuals of the Eclipse model indicate that the model adheres well to assumptions critical for its validity and reliability in time-series forecasting.

* **Shapiro-Wilk Test (Statistic = 0.9819, p-value = 0.32554):** This test evaluates whether the residuals follow a normal distribution. The p-value of 0.32554 is greater than the typical significance level of 0.05, indicating no significant departure from normality. This suggests that the residuals are approximately normally distributed, which supports the reliability of the model’s predictions.
* **Kolmogorov-Smirnov Test (Statistic = 0.0439, p-value = 0.99645):** This test further assesses normality by comparing the residuals to a normal distribution. The extremely high p-value of 0.99645 confirms that the residuals align closely with a normal distribution, reinforcing the findings from the Shapiro-Wilk test.
* **Durbin-Watson Test (Statistic = 1.9360):** This statistic tests for autocorrelation in the residuals. A value close to 2 (specifically, 1.9360) indicates minimal autocorrelation, confirming that the residuals are independent. This is crucial for time-series models, as residual independence ensures unbiased predictions and validity of confidence intervals.
* **ADF Test (Statistic = -8.4595, p-value = 1.5810e-13):** The Augmented Dickey-Fuller test checks for stationarity in the residuals. The highly negative test statistic (-8.4595) and the p-value of 1.5810e-13 (far below 0.05) strongly reject the null hypothesis of non-stationarity. This result confirms that the residuals are stationary, meaning their statistical properties (e.g., mean and variance) remain constant over time.
* **Critical Values for ADF Test:** The test statistic (-8.4595) is significantly lower than the critical values at the 1% (-3.5171), 5% (-2.8994), and 10% (-2.5870) levels, further confirming the stationarity of the residuals.

These statistical tests collectively confirm that the residuals of the Eclipse model satisfy key assumptions:

1. They are approximately normally distributed (Shapiro-Wilk and Kolmogorov-Smirnov tests).
2. They exhibit minimal autocorrelation (Durbin-Watson test).
3. They are stationary (ADF test).

A screenshot of a data

Description automatically generated

* 1. Additional analysis

**Training and Validation MAE**

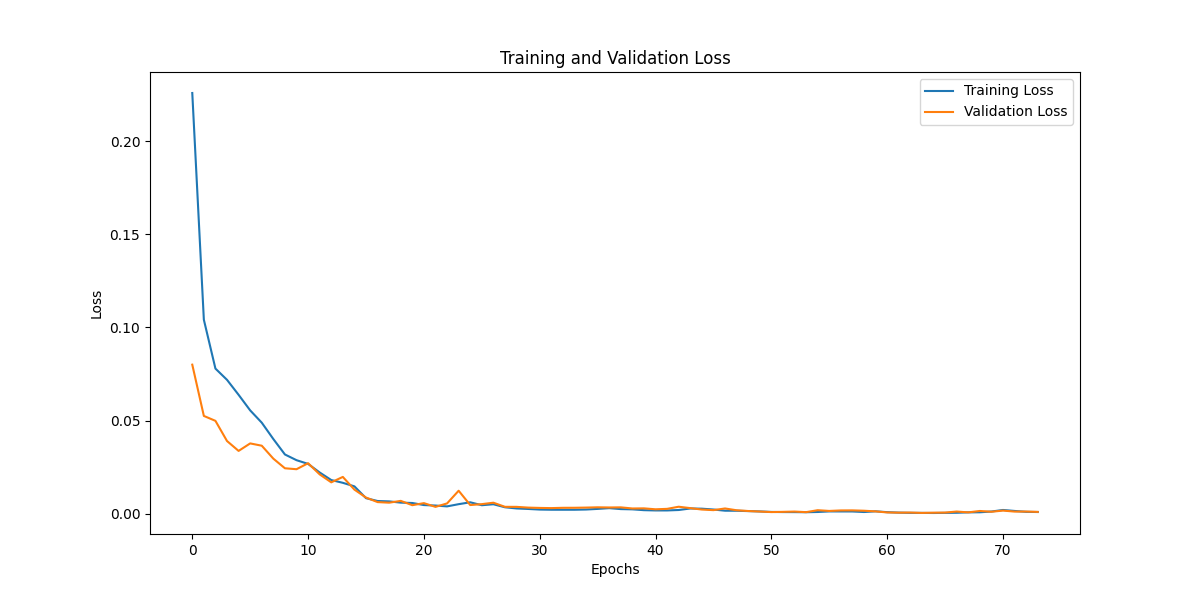
The plot demonstrates the Mean Absolute Error (MAE) trends for both the training and validation sets across epochs. Initially, both curves exhibit higher MAE values, indicating greater prediction errors, but they rapidly converge and stabilize as the model learns from the data. The validation MAE closely follows the training MAE without significant divergence, suggesting minimal overfitting and strong generalization to unseen data.

A graph with blue and orange lines

Description automatically generated

**Training and Validation Loss**

The loss curves indicate the progression of the model's learning process. Both training and validation losses decline steeply in the early epochs and plateau as the model converges. The near-parallel nature of the curves and their stabilization at similar levels highlight that the model effectively balances fit and generalization. The absence of divergence between training and validation loss reaffirms that the model avoids overfitting.



**Feature Correlation Matrix**

A screenshot of a computer screen

Description automatically generatedHigh positive correlations (e.g., MVRVZ, NUPL) suggest that these variables significantly influence the dependent variable, while negative correlations (e.g., MA Ratio) provide insights into inversely related factors. The matrix validates the feature selection process by confirming that the selected variables capture diverse and meaningful relationships with the target. It also highlights potential multicollinearity (e.g., high correlations between Shifted CVDD Background Ratio and MVRVZ), which the LSTM architecture handles well.

**Histogram of Residuals (Validation Set)**

A graph of a graph

Description automatically generated with medium confidenceThe symmetric, bell-shaped curve centered around zero suggests that the residuals are approximately normally distributed, as confirmed by the Shapiro-Wilk and Kolmogorov-Smirnov tests. The absence of significant skewness or extreme outliers indicates that the model produces unbiased predictions with errors distributed evenly.

**Residuals vs Predictions**

The scatterplot of residuals against predictions confirms the absence of systematic bias in the model’s predictions. The residuals are randomly scattered around the zero-error line, with no discernible patterns or trends, suggesting that the errors are independent of the predictions. This supports the validity of the model and ensures that no key relationships in the data were overlooked during training.

A graph of a graph showing a line

Description automatically generated with medium confidence

**True vs Predicted Values (Validation Set)**

A graph with orange and blue lines

Description automatically generatedThis graph compares the true values of the dependent variable with the model’s predictions for the validation set. The near-perfect overlap of the predicted curve (dashed line) with the true values indicates that the model effectively captures the underlying patterns in the data. The minor deviations suggest small residual errors, which are expected but remain well within acceptable ranges.

1. Discussion

While the Eclipse Indicator demonstrates robust predictive performance based on historical data and shows strong generalization capabilities, it is not without limitations. One significant constraint is its reliance on past patterns and relationships within the Bitcoin market. The model assumes that these relationships remain stable over time; however, markets, especially emerging ones like Bitcoin, are subject to significant structural changes. Factors such as regulatory developments, macroeconomic influences, and shifts in investor behavior can disrupt previously observed patterns, potentially reducing the accuracy of predictions.

Additionally, the model’s reliance on selected features—while well-correlated with the dependent variable—means it is limited by the completeness and quality of the input data. Unexpected changes in market dynamics or the introduction of new, impactful variables that are not part of the current feature set may reduce the indicator's reliability. Moreover, the inherent complexity of deep learning models like LSTMs can sometimes obscure interpretability, making it challenging to identify the specific reasons for prediction errors when they occur.

Given that the Bitcoin market is still emerging and characterized by rapid evolution, the Eclipse Indicator is expected to lose precision over time as market conditions deviate from the historical patterns it was trained on. To address this, the model will be systematically updated after each newly confirmed market top or bottom. This process ensures that the indicator remains aligned with the latest data and incorporates any structural changes in the market. By recalibrating the model periodically and redefining extreme values based on updated conditions, the Eclipse Indicator can maintain its relevance and accuracy as an essential tool for understanding Bitcoin's cyclical behavior.

1. Références
   1. Crédits tradingview & contributeurs

<https://charts.bitbo.io/fear-greed/>

<https://www.coinglass.com/fr/pro/i/bitcoin-rainbow-chart>