Regime-Aware Anomaly Detection in Cryptocurrency Wallet Clusters: Uncovering Coordinated Behavior in Markets

Guillermo Marr

September 4, 2025

1 Introduction

Cryptocurrency interest has soared in recent years, the next gimmick to make quick money. However, cryptocurrency markets are very volatile and involve lots of unregulated activity. This unregulated activity leads to cases of market manipulation, rug pulls, pump and dumps, etc. Retail traders often are at the short end of the stick of these attacks. There have been attempts in trying to detect malicious behavior in the market, such as whale movements or synchronized fund shifts, that trigger crises or rapid price change. However these traditional methods focus on flagging singular irregularities such as unusual price spikes or suspicious transactions. While effective in some ways, these approaches fail to take into account the markets regime, leading to potential false positives (flagging big sell off as anomaly in a bear market) or missed detections of coordinated attacks. This is a disservice to the problems they aim to solve, as the best way to understand the market is to look at the psychology of the market and where the money is at. If people are confident a price will go somewhere the price would have already gone there, people lie, money doesn't.

This report covers a new approach, that is, to incorporate regime-aware anomaly detection framework to fix these issues, showcased on simulated market data. The central research question looked to answer in this paper is Does regime awareness improve a models ability to identify anomalies in the market, and reduce false positives? By allowing the model to have market context (bear, bull, neutral) we hypothesize the model will be able to distinguish normal market psychology from true anomalies in the market, reducing false alarms, and improving accuracy for significant shifts. The idea is that by allowing the model to know the state it is in, the parameters in which we cluster are adjusted to be more useful. For example, in a bear market large amounts of sell orders should not be flagged, as those are common in bear markets. In contrast an ignorant model that has no market context may classify a large sell off as an anomaly and therefore act in accordance to this false information.

I plan to have a robust methodology to test my research question, which consists of the following: Hidden Markov Model (HMM) for detecting regimes, DBSCAN for grouping similar wallets together based on their patterns, and Isolation Forest for detecting anomalous clusters or market groups. This methodology is applied to synthetic cryptocurrency coin transaction data, and compared to a non regime-aware approach to evaluate the performance improvement of regime awareness.

2 Literature Review

Financial markets has seen tremendous amounts of attempts for detecting anomalies. The purpose of these works can be for the following; risk detection, malicious transaction detection, and for the interest of making money. With growing relevance in cryptocurrencies, especially meme coins like WIF, PEPE, DOGE, and SHIBA, the need for robust methodologies in detecting anomalies is growing, as their volatility and lack of governance makes them a target for manipulation. This is mainly due to blockchains transparency and complexity [3]. Recent works have tested the boundaries of this field by utilizing different machine learning techniques, but persistent gaps remain.

Studies have explored machine learning based anomaly detection in cryptocurrency, each with unique insight but falling short in different ways. One paper, ”Exploring Machine Learning-Based Methods for Anomaly Detection: Evidence from Cryptocurrencies” [1], investigated anomalies in Bitcoin and Dogecoin prices using the folloiwng: Local Outlier Factor(LOF), Isolation Forest(IF), and One-Class Support Vector machine (OCSVM). It was shown that LOF was the most effective method for detecting outliers in cryptocurrency returns. However the study focused on price data only, limiting the applicability to transaction level anomalies, missing very crucial information where manipulations often unfold through coordinated wallet activities rather than just change in price. It needs to focus on WHAT is moving the price, not the price itself.

Another paper relative to the prior, is ”Detecting Anomalies in Blockchain Transactions Using Machine Learning Classifiers and Explainability Analysis” [2] which used classifiers to identify suspicious Bitcoin transactions and pairing it with explainability (XAI) techniques into tree ensemble methods. The approach gave information into why a model flagged a certain transaction, and gave great analysis into the black box of the model. This method was very creative and novel however it did not contextualize anomalies within the broader market, which makes it less relevant to real world applications. In volatile markets such as cryptocurrency markets, the need for modeling behavior is heavily needed, because a flip in the market volatility can flip any second depending on the markets state, which was an oversight in this paper.

Now we hone in on a more niche paper, ”Machine Learning-Based Detection and Analysis of Suspicious Activities in Bitcoin Wallet transactions in the USA” [5] which looked into Bitcoin wallet data specifically in the US. The authors used logistic regression, random forest, and support vector machines, and identified trends that could showcase fraud. While these methods were effective for real-time detection in a specific region (US), the price and movement of a cryptocurrency token is dependent on global valuation and not just what is seen on one exchange. The limit of this paper on a global scale to capture the full volatile nature of the markets, because anomalies requires more information incorporated due to potential region spread attacks.

A more advanced approach showcased in the paper ”A GNN-based Graph Classification Framework for MEV Activity Detection” [6], took raw transactional data and translated it into a user behavior graph model, by employing a graph neural network (GNN) to figure out when Maximum Extractable Value (MEV) activity was taking place on decentralized networks. This paper showed the potential of deep learning has for identifying and enhancing blockchain security and fairness. However, the complexity computational and the need for labeled data are some real big challenges for broader application of this approach, especially in environments where data is not as available.

Lastly we will look at a paper ”The Kosmosis Use-Case of Crypto Rug Pull Detection and Prevention" [7] which addressed the need for rug pull detection mechanisms by using a knowledge graph of blockchain and social media data. This multi modal approach of using alternate data brings great insight into detecting manipulative schemes fueled by hype. However, the focus is on rug pulls a specific type of fraud, and may not generalize to real time transaction anomaly detection, due to the lag of social media sentiment data having high variance.

Despite these papers great achievements and demonstration in the markets, there lies a critical gap: which is that few studies explicitly add regime awareness into their model, particularly for cryptocurrency problems. Market regimes (bull, bear, neutral) shape the transaction behavior in the market tremendously. For example in a bull market, high buy volume does not reflect manipulation but rather most likely market psychology of retail traders buying off hype. Existing methods are robust in their contexts but treat anomalies as isolated events, they ignore the markets mood. This leaves so much information out of context and may serve as a tremendous hinder to the growth of accuracy of the model.

3 Methodology

Our proposed regime-aware anomaly detection framework consists of three parts:

1. **Regime Detection** via Hidden Markov Models (HMMs) to split the market into distinct regimes (bull, bear, neutral).
2. **Clustering** of wallets via their features within each regime using DBSCAN, an algorithm very useful for putting data into groups and separating them.
3. **Anomaly Detection** for finding anomalous groups in our clusters for a certain regime, indicating likely market manipulation or bad actors in that group.

We showcase the mathematical foundation behind each process and provide the intuition behind how they work together to reduce false positives and improve detection of coordinated activity in cryptocurrency markets.

3.1 Regime Detection: Hidden Markov Model (HMM)

Accurate detection of market regimes (e.g., bull, bear, or neutral states) is a must for adjusting the parameters in our other models. This makes sense intuitive because different markets behavior differently, and it is due to the psychology behind each state. By modeling historical data with an HMM we are able to exploit the underlying structure that our market is in and allows us to better understand the impact of certain groups.

3.1.1 HMM Intuition and Use in Regime Detection

A Hidden Markov Model is especially well-suited for capturing regime dynamics because it:

- *Maintains discrete hidden states* (the regimes), which are not known but implied based on log returns and volatility of the market.
- *Captures Markovian state transitions*—this is fundamental to many problems and it is the state of the next regime only depends on the current regime not the prior history of regimes.

- *Learns observation likelihoods*—the distribution of returns is probabilistic, same with volatility, and it is learned through training.

In simpler terms, each hidden state (bear, neutral, bull) is a response to a certain distribution of returns r_t and volatility σ_t. The ever changing evolution of the states is done by a state transition matrix.

3.1.2 Mathematical Specification

States and Observations. An HMM is formally specified by:

$$\lambda = (S, O, A, B, \pi),$$

where:

- $S = \{\text{bear}, \text{neutral}, \text{bull}\}$ is the set of hidden states.
- $O = \{O_1, O_2, \dots, O_T\}$ is the sequence of observations, where $O_t = (r_t, \sigma_t)$ (e.g., returns and volatility at time t).
- $A = [A_{ij}]$ is the transition matrix, with elements

  $$A_{ij} = P(S_t = j \mid S_{t-1} = i),$$

  indicating the probability of moving from state i to state j.
- $B = \{B_j(\cdot)\}$ are the emission probability distributions, where

  $$B_j(o) = P(O_t = o \ \mid \ S_t = j).$$
- $\pi = [\pi_i]$ is the initial state distribution, where

  $$\pi_i = P(S_1 = i).$$

Likelihood of Observations. Given a sequence of observations $\mathbf{O} = (O_1, O_2, \dots, O_T)$, the likelihood under the HMM is

$$P(\mathbf{O} \mid \lambda) \ = \ \sum_{S_1, \dots, S_T} P(S_1, \dots, S_T, \, O_1, \dots, O_T \mid \lambda).$$

This summation, taken over all possible hidden state sequences $(S_1, \dots, S_T)$, can be computed efficiently via the *forward-backward* algorithm.

Forward-Backward Algorithm. Define the forward variable $\alpha_t(i)$ as

$$\alpha_t(i) = P(O_1, O_2, \dots, O_t, \ S_t = i \mid \lambda),$$

and the backward variable $\beta_t(i)$ as

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \dots, O_T \ \mid \ S_t = i, \, \lambda).$$

Then:

$$P(\mathbf{O} \mid \lambda) = \sum_i \alpha_T(i).$$

The recursions for $\alpha_t(i)$ and $\beta_t(i)$ are given by:

Forward step:

$$\alpha_1(i) = \pi_i \, B_i(O_1),$$

$$\alpha_{t+1}(j) = \left[\sum_i \alpha_t(i) \, A_{ij} \right] B_j(O_{t+1}), \quad t = 1, \dots, T-1,$$

Backward step:

$$\beta_T(i) = 1,$$

$$\beta_t(i) = \sum_j A_{ij} \, B_j(O_{t+1}) \, \beta_{t+1}(j), \quad t = T-1, \dots, 1.$$

Parameter Re-estimation (Baum-Welch). The HMM parameters $\lambda = (A, B, \pi)$ are updated to maximize the likelihood $P(\mathbf{O} \mid \lambda)$ using an Expectation-Maximization (EM) procedure known as *Baum-Welch*. One key set of updates is for the transition matrix A_{ij}:

$$A_{ij} \leftarrow \frac{\sum_{t=1}^{T-1} \gamma_t(i,j)}{\sum_{t=1}^{T-1} \sum_{j'} \gamma_t(i,j')},$$

where

$$\gamma_t(i,j) = \frac{\alpha_t(i) \, A_{ij} \, B_j(O_{t+1}) \, \beta_{t+1}(j)}{P(\mathbf{O} \mid \lambda)}.$$

The emission distribution parameters $B_j(\cdot)$ and the initial state distribution π are updated similarly. Upon convergence, the *Viterbi algorithm* is applied to find the single most likely sequence of hidden states $\{S_t\}$.

Interpretation for Crypto Regimes. In our setting, the hidden states $S = \{\text{bear}, \text{neutral}, \text{bull}\}$ characterize different market regimes, each with distinct statistical significance for returns (r_t) and volatilities (σ_t). The estimated transition probabilities A_{ij} indicate how likely it is for the market to remain in (or switch to) each different regime, this allows for insight into the the persistence or switching at play in our market cycles.

3.2 Clustering: DBSCAN

3.2.1 DBSCAN Foundations

Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ be a set of points with distance function $d(\mathbf{x}, \mathbf{y})$. DBSCAN requires two parameters: $\epsilon > 0$ and $MinPts \in \mathbb{N}$. Define:

$$N_\epsilon(\mathbf{x}) = \{\mathbf{y} \in X \mid d(\mathbf{x}, \mathbf{y}) \leq \epsilon\}.$$

A point $\mathbf{x}$ is:

- **Core point** if $|N_\epsilon(\mathbf{x})| \geq MinPts$.
- **Border point** if $|N_\epsilon(\mathbf{x})| < MinPts$ but $\mathbf{x}$ is reachable from some core point.
- **Noise** if it is neither a core nor border point (DBSCAN labels it as -1).

Direct Density-Reachability: $\mathbf{x}$ is directly density-reachable from $\mathbf{y}$ if $\mathbf{y}$ is a core point and $\mathbf{x} \in N_\epsilon(\mathbf{y})$. Clusters form by chaining together density-reachable points.

3.2.2 Regime-Aware Parameter Tuning

We let $(\epsilon, MinPts)$ change by market regime (e.g., bear, bull, neutral) to account for different behavior in different regimes. In high-volatility (bull) regimes, ϵ is smaller to avoid merging irregular clusters; in low-volume (bear) regimes, ϵ can be larger to capture bigger but legitimate activity.

3.2.3 Wallet-Level Features

Instead of clustering raw transactions, each wallet w in regime r is mapped to a feature vector $\mathbf{f}_{w,r}$ (e.g., total sell volume, frequency, time concentration). DBSCAN on these $\mathbf{f}_{w,r}$ helps reveal coordinated patterns. We want to focus on the wallets and their behavior, not individual noise in the transactions.

3.3 Anomaly Detection: Isolation Forest

3.3.1 Isolation Forest Fundamentals

Given n points $\{\mathbf{z}_1, \ldots, \mathbf{z}_n\}$ in a feature space, Isolation Forest builds T random trees. Each tree selects a feature index, chooses a split between min and max of that feature and partitions until each point is isolated.

The *isolation depth* $h(\mathbf{z})$ is the path length from root to the leaf isolating $\mathbf{z}$. The **isolation score** $s(\mathbf{z})$ is:

$$s(\mathbf{z}) = 2^{-\frac{h(\mathbf{z})}{c(n)}},$$

where $c(n)$ is a normalization factor (e.g., $c(n) \approx 2\, H_{n-1} - \frac{2(n-1)}{n}$, with H_{n-1} the $(n-1)$-th harmonic number). Higher $s(\mathbf{z})$ indicates greater anomaly.

3.3.2 Regime-Specific Contamination

Let ν_r be the *contamination* parameter in regime r. If suspicious activity is more common in, for example, bull markets, we set $\nu_{\text{bull}} > \nu_{\text{bear}}$. This is to make sure that anomalies are correctly identified in high anomalous regimes.

3.3.3 Combining DBSCAN and Isolation Forest

We apply DBSCAN to obtain cluster labels or wallet-group aggregates, then feed them into an Isolation Forest:

1. **Cluster-Level:** Compute cluster-level statistics (size, total volume, etc.) → isolate suspicious clusters.

2. **Wallet-Level:** Directly compute anomaly scores for each wallet's feature vector, using cluster labels as auxiliary features if needed.

This synergy captures subtle group behaviors (e.g., synchronized whale sells) while remaining robust to normal transaction patterns under different regimes.

4 Experimental Setup

We evaluate our **Regime-Aware Anomaly Detection** on synthetic data and compare to a *baseline* model that ignores regime distinctions (i.e., uses fixed parameters) and see if we there is improvement.

4.1 Dataset: Synthetic Coin Transactions

Transactions. We simulate 1,000 wallets executing 50,000 transactions over 30 days, randomly splitting the daily volume to reflect periods of *Bull*, *Bear*, or *Neutral* states. The transactions include:

- *Amount (in tokens)*—sampled from distributions that differ by regime.
- *Timestamp*—uniformly drawn within each hour.
- *Counterparties*—wallets traded between.

Price and Volatility. We generate 720 hourly data points for the crypto coin's price and volatility, using an HMM with 3 hidden states to produce synthetic log-returns r_t and volatilities σ_t.

Anomalies. We inject wallets exhibiting *coordinated behavior*: large and frequent sells in tight windows. This simulates "whales" attempting to manipulate or coordinate large sells to front-run momentum or create panic in the market for their own gain.

4.2 Methodological Steps in Implementation

1. **Regime Detection (HMM):** Train an HMM on price returns (r_t) and volatility (σ_t). The hidden states are labeled as bear, neutral, bull match each states average return and voltaility to a regime.
2. **Clustering (DBSCAN):** For wallets that fall into each regime's time window, extract their transactional features (e.g., total net volume, frequency, average transaction size). We then run a separate DBSCAN instance grouping wallets by these features with regime-specific ($\epsilon, MinPts$).
3. **Anomaly Detection (Isolation Forest):** Finally, we get the cluster features from each cluster and run Isolation Forest on this data. The contamination parameter ν is set higher in bull/neutral states to accommodate more frequent bursts of unusual activity, while in bear states we lower it to reduce false positives from low-volume noise.
4. **Baseline (Static Parameters):** As a comparison, we also run a single DBSCAN + single Isolation Forest *without* regime separation.

4.3 Evaluation Metrics

- **Anomaly Detection Accuracy:** Evaluate **Precision**, **Recall**, and **F1-score** for identifying the injected wallets.
- **False Positive vs. True Positive Rates:** We track how many normal wallets are incorrectly flagged (FPR) and how many anomalies are correctly flagged (TPR) in the regime-aware vs. non-regime-aware approach.

5 Results

For our regime-aware method we observed:
Precision: 51.7
Recall: 38.5
F1: 44.1
For the non-regime baseline using a static approach the evaluation yielded:
Precision: 20.5
Recall: 23.1
F1: 21.7

In comparing the two approaches, the regime-aware model significantly outperforms the static non-regime baseline on Precision and F1. The improved precision of the regime-aware method suggests that by partitioning transactions according to market regime and then computing wallet-level features and cluster-level statistics separately, the method is better able to differentiate the coordinated anomalous behavior from normal activity. This backs up our hypothesis that regime awareness helps aid the model, even if its slight adjustment as shown in this paper. By training the DBSCAN clustering and Isolation Forest on each regime's distinct features, the framework is capable of capturing subtle differences that are lost when aggregating all transactions together (as in the non-regime baseline).

6 Conclusion and Future Directions

In summary, **regime-aware anomaly detection** is better than a static approach for identifying suspicious behavior in the market, especially in *cryptocurrency* contexts where hype-driven, high-volatility events can confound traditional methods. By separately learning parameters for *bear*, *neutral*, and *bull* regimes, our framework adapts to market conditions and achieves higher detection accuracy on synthetic data.

Potential Extensions: To further improve our work and for future papers, it would be useful to use high order HMMs, due to crypto markets potential to be rapidly changing and be a lot different than their first order Markov transitions. Higher-order HMMs could capture this potentially. Real time implementation as well would be useful to add, to update regime labels hour by hour then adjusting DBSCAN and Isolation Forest in near real time.

Overall, the synergy of **Hidden Markov Models**, **DBSCAN clustering**, and **Isolation Forests** is an effective methodology for balancing adaptability and detection power.

References

[1] Yahia, A., Mouhssine, Y., El Alaoui, A., et al. (2024). Exploring machine learning-based methods for anomalies detection: evidence from cryptocurrencies. *International Journal of Data Science and Analytics.* https://doi.org/10.1007/s41060-024-00703-w

[2] Hasan, M., Rahman, M. S., Janicke, H., & Sarker, I. H. (2024). Detecting anomalies in blockchain transactions using machine learning classifiers and explainability analysis. *Blockchain: Research and Applications*, 5(3), 100207. https://doi.org/10.1016/j.bcra.2024.100207

[3] Zheng, Z., et al. (2020). Blockchain intelligence: When blockchain meets artificial intelligence. *arXiv preprint arXiv:1912.06485.* https://arxiv.org/abs/1912.06485

[4] Yıldız, K., Dedebek, S., Okay, F. Y., & Şimşek, M. U. (2022). Anomaly detection in financial data using deep learning: A comparative analysis. In *2022 Innovations in Intelligent Systems and Applications Conference (ASYU)* (pp. 1-6). IEEE. https://doi.org/10.1109/ASYU56188.2022.9925392

[5] Islam, M. Z., Islam, M. S., Das, B. C., Reza, S. A., Bhowmik, P. K., Bishnu, K. K., Rahman, M. S., Chowdhury, R., & Pant, L. (2025). Machine learning-based detection and analysis of suspicious activities in Bitcoin wallet transactions in the USA. *Journal of Ecohumanism*, 4(1), 3714. https://doi.org/10.62754/joe.v4i1.6214

[6] Yao, Z., Huang, F., Li, Y., Duan, W., Qian, P., Yang, N., & Susilo, W. (2025). Mecon: A GNN-based graph classification framework for MEV activity detection. *Expert Systems with Applications*, 269, 126486. https://doi.org/10.1016/j.eswa.2025.126486

[7] Stangl, P., & Neumann, C. P. (2024). The Kosmosis use-case of crypto rug pull detection and prevention. *arXiv preprint arXiv:2405.19762.* https://arxiv.org/abs/2405.19762. Accessed 24 Feb. 2025.