International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Protecting Investors from Meme Coin Scams: A Smart Analysis Tool

Naman[1], Surya Gupta[2], Shivani Chourey[3]

[1,2,3]Department of Computer Science, India

Abstract

Meme coins are incredibly risky investments in the cryptocurrency world. While they get popular through social media hype and viral communities, many turn out to be scams where developers steal investor money and disappear—what's known as a "rug pull." We built an intelligent system that analyzes both the code and market behavior of meme coins to warn investors before they lose their money. Our tool checks things like who controls the coin, how the money is locked up, and whether the developers are acting suspiciously. After testing it on real meme coins, both legitimate ones and proven scams, our system accurately identifies which projects are dangerous.

1 Why This Matters

1.1 The Problem with Meme Coins

Cryptocurrency has made it possible for anyone to create digital money without ask- ing permission from banks or governments. Meme coins—cryptocurrencies inspired by internet jokes and cultural trends—have become wildly popular thanks to celebrity en- dorsements and speculative trading. Unlike Bitcoin or Ethereum, most meme coins don't actually do anything useful. They're valuable only because people believe they are, driven by social media buzz and community excitement.

Creating a token on platforms like Ethereum or Binance Smart Chain is surprisingly easy and cheap. This accessibility is great for innovation, but it's also made it simple for scammers to launch fake projects. They can create what looks like a legitimate cryptocurrency, get people to invest, then steal all the money within days or even hours. Victims are left holding worthless tokens.

1.2 Why Scams Are Hard to Spot

The tricky part is that scam projects often look completely legitimate at first. Clever scammers will lock up liquidity temporarily to build trust, spread tokens across multiple wallets to fake organic growth, or even give up their ownership rights to appear trustwor- thy. These deception tactics show why we need ongoing, multi-layered analysis rather than just checking if something looks safe at launch.

Most existing security tools only look at code vulnerabilities or use simple red flags like checking if the developer still owns the contract. But this narrow approach misses the bigger picture, especially for meme coins which have unique risk patterns. The fast- moving nature of crypto markets, combined with the psychological manipulation tactics scammers use, requires a more complete analytical approach.

1.3 Our Contribution

We demonstrate that meme coins have measurable characteristics that can be analyzed

systematically. Our automated system evaluates multiple risk factors: who has special permissions in the code, whether investor money is properly locked up, how the tokens are distributed among holders, and what the developers are doing with their wallets. By com- bining rule-based scoring with machine learning, we give investors clear risk assessments that can prevent significant financial losses.

2 What Others Have Done

Previous blockchain security research has mostly focused on general smart contract bugs rather than detecting scams specifically. Vasek and Moore [1] studied Ponzi schemes in Bitcoin, showing that blockchain transparency allows us to analyze fraud patterns after the fact, though catching scams in real-time remained difficult.

Chen and colleagues [2] developed systems to detect scam tokens on Ethereum and Binance Smart Chain by analyzing transaction patterns. Their approach found suspicious trading behaviors but didn't fully examine tokenomics or smart contract permissions. We build on this foundation by integrating multiple analytical perspectives specifically designed for meme coin characteristics.

The rise of automated market makers and liquidity pools has created new ways for scammers to operate that earlier research didn't address. Our framework specifically targets these modern DeFi vulnerabilities while remaining compatible with traditional security analysis methods.

3 How Our System Works

3.1 Overall Design

Our Meme Coin Fundamentals Analyzer uses a modular pipeline architecture that com- bines blockchain data extraction, feature engineering, and automated risk evaluation. The system is transparent and can be deployed as a web app, browser extension, or API service. This versatility means it can integrate into existing DeFi tools and wallets to provide real-time risk assessment when users are evaluating tokens.

The architecture has three main layers working together:

1. **On-Chain Smart Contract Analysis**: Examines the deployed code to find po- tentially malicious permissions or hidden functions
2. **Off-Chain Tokenomic Analysis**: Gathers data from blockchain explorers and liquidity pools to assess token distribution and market structure
3. **Trust Evaluation Engine**: Combines inputs from both layers to generate a final risk score and classification

3.2 Component Details

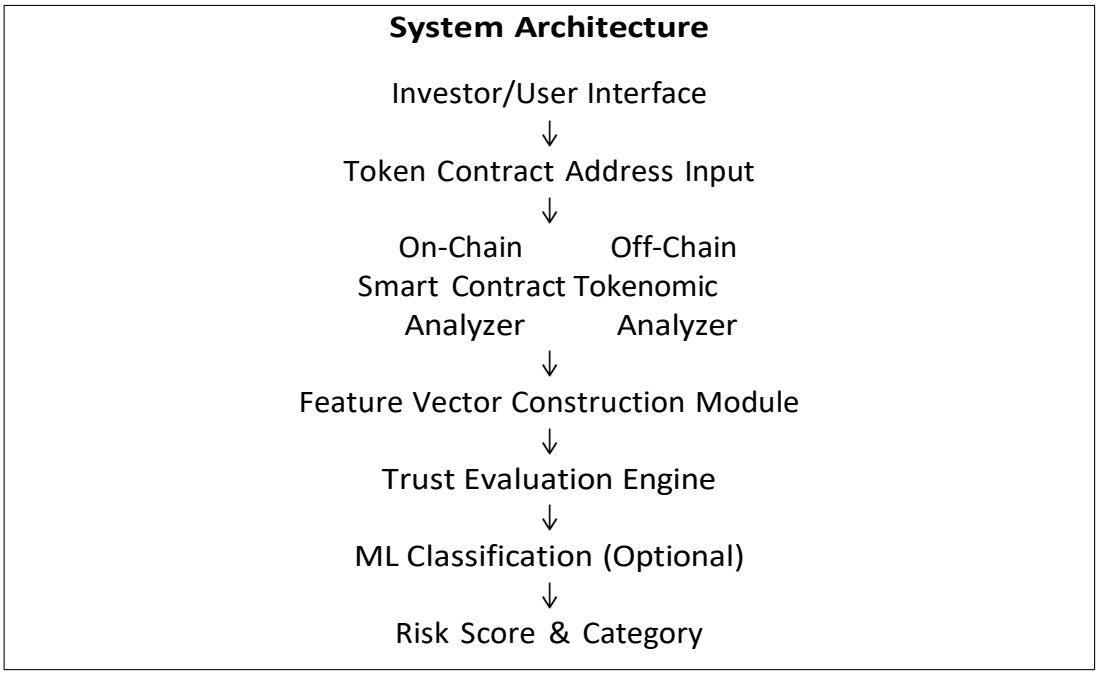

Figure 1: High-level system architecture showing the modular pipeline design of the Meme Coin Fundamentals Analyzer with data flow between components

Figure 1 illustrates the complete data flow through the system architecture. User input triggers parallel analysis pipelines that reconverge at the feature vector construction stage, ensuring comprehensive evaluation before final scoring.

The On-Chain Analyzer connects directly to blockchain nodes to retrieve contract code, function definitions, and transaction histories. It uses specialized parsers for token standards and their variants, enabling detection of hidden administrative functions that might not be documented publicly. The analyzer checks whether ownership has been given up, looks for hidden minting or burning capabilities, checks for pause mechanisms, and identifies unfair transaction taxes that could trap investor funds.

The Off-Chain Analyzer pulls data from multiple sources including DEX aggrega- tors, blockchain explorers, and liquidity locker services. It calculates holder distribution metrics using inequality measures, evaluates liquidity pool depth and lock duration, and tracks developer wallet activity. By monitoring multiple data streams simultaneously, the system can detect coordinated manipulation attempts that might look innocent when ex- amined separately.

The Trust Evaluation Engine uses both rule-based scoring and optional machine learning classification. The rule-based component applies weighted scoring across nor- malized features, ensuring users can understand which specific factors contribute to the final risk assessment. The machine learning component uses ensemble methods trained on historical data to identify subtle patterns that might indicate sophisticated scam op- erations.

4 Our Analytical Methods

4.1 Analyzing Smart Contracts

We analyze smart contracts using a combination of interface decoding and bytecode in- spection. The

process starts by retrieving the contract's verified source code or deployed bytecode from the blockchain. For verified contracts, we parse the code to identify func- tion signatures, access modifiers, and state variables. For unverified contracts, we decom- pile the bytecode to reconstruct probable functionality.

The ownership analysis checks whether the contract has standard ownership patterns and whether ownership has been properly given up. Many meme coin contracts include owner-only functions that let privileged addresses change critical parameters like trans- action fees, maximum transaction amounts, or trading on/off switches. Unrelinquished ownership is a major red flag because owners can exploit these privileges to manipulate the token or prevent people from selling.

Hidden function detection identifies potentially malicious capabilities that aren't im- mediately obvious. This includes searching for mint functions that could inflate supply, burn functions that might artificially reduce circulation, and pause mechanisms that could freeze all transactions. We maintain a database of known malicious function patterns from previous scam contracts.

Transaction tax asymmetry detection examines buy and sell tax implementations to find cases where sell taxes are significantly higher than buy taxes or where taxes can be changed arbitrarily. Such asymmetries can trap investor funds by making selling economically impossible.

4.2 Analyzing Token Economics

Off-chain analysis uses blockchain explorer APIs and DEX protocol interfaces to extract comprehensive market and distribution data. This provides context that contract code inspection alone cannot reveal, showing how the token is actually being used and traded in practice.

Liquidity pool analysis evaluates the depth, stability, and security of token liquidity. We query DEX contracts to determine total liquidity value, the ratio of token reserves to base currency reserves, and who is providing liquidity. Critically, we check whether liquidity tokens are locked in verified time-lock contracts and calculate the remaining lock duration. Unlocked or briefly locked liquidity represents severe risk because developers can remove it anytime, instantly making the token untradeable.

Holder distribution analysis calculates statistical measures of token concentration across wallet addresses. We retrieve the complete list of token holders and their bal- ances, then compute metrics like the percentage of supply held by top holders, inequality coefficients, and the number of unique holders. High concentration in developer-controlled wallets suggests potential coordinated dumping risk.

Developer wallet tracking identifies addresses associated with the project team and monitors their transaction patterns. Sudden large transfers from developer wallets to ex- changes often precede rug pulls. We maintain a watchlist of identified developer addresses and generate alerts when suspicious activity is detected.

4.3 Processing the Data

Raw metrics from on-chain and off-chain analysis are processed through a feature en- gineering pipeline that normalizes values, handles missing data, and constructs derived features. Normalization ensures that features with different scales contribute proportion- ally to the final score. For example, liquidity lock duration measured in days is normalized to a 0-1 scale.

Derived features capture relationships between multiple raw metrics. For instance, the ratio of developer-held tokens to total liquidity provides insight into potential price impact if developers decide to sell. The relationship between holder count and trading volume can reveal artificial activity

inflation through bot networks.

4.4 Calculating Trust Scores

The trust scoring mechanism combines normalized features using a weighted linear model designed for clarity. Each feature is assigned a weight based on its observed correlation with historical rug-pull events. Features with stronger predictive power receive higher weights.

The weighted sum produces a composite trust score ranging from 0 (maximum risk) to 100 (minimum risk). Threshold values divide this continuous score into discrete risk categories: Critical Risk (0-30), High Risk (31-50), Moderate Risk (51-70), and Low Risk (71-100). These categories enable quick decision-making while preserving the underlying continuous assessment for users who want more detail.

4.5 Using Machine Learning

An optional machine learning component enhances the rule-based scoring by learning non- linear relationships and interaction effects between features. We use ensemble methods including Random Forest and Gradient Boosting classifiers trained on labeled historical data. The ML model outputs a probability estimate that complements the rule-based trust score.

The machine learning pipeline includes cross-validation to prevent overfitting and regular retraining on updated data to adapt to evolving scam tactics. Feature importance analysis from the trained models reveals which characteristics most strongly indicate fraudulent projects.

Figure 2 visualizes the decision flow and data processing stages. The branching logic allows the system to operate in multiple modes depending on user preferences and avail- able computational resources.

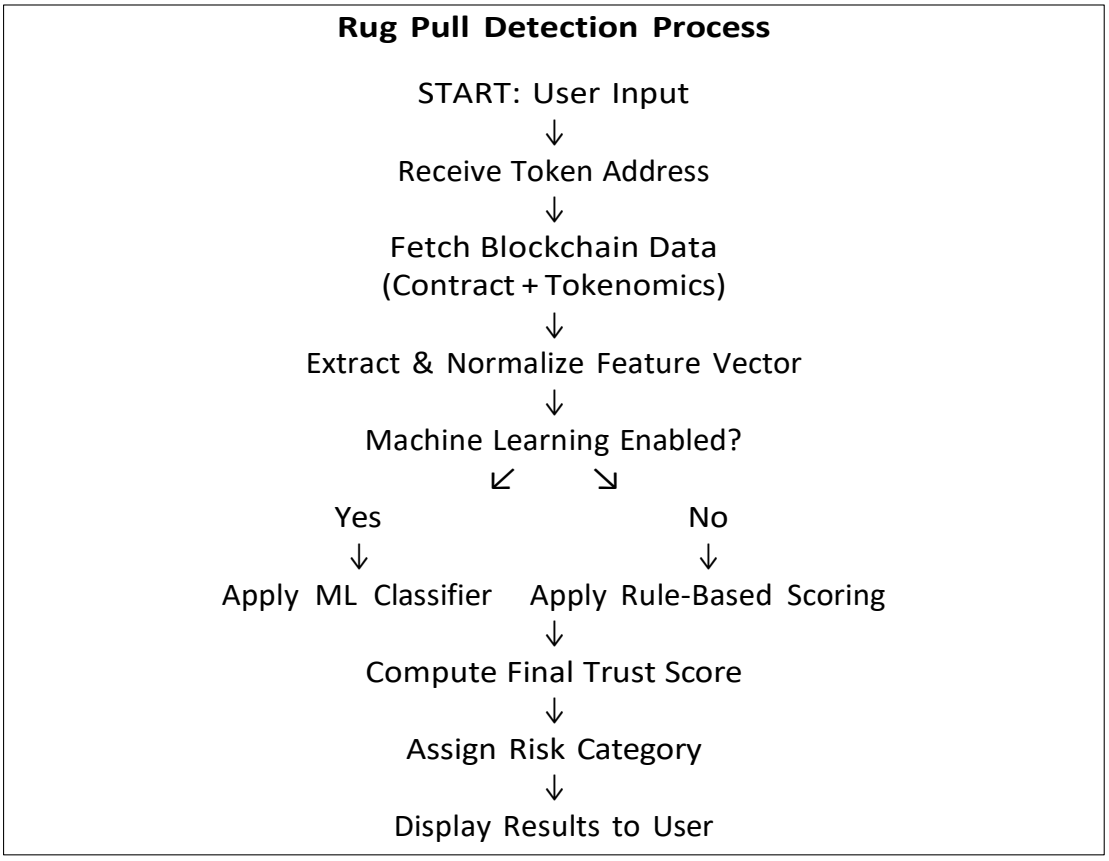

Figure 2: Detailed flowchart illustrating the complete rug-pull detection process from user input through final risk categorization

5 Our Test Data

The evaluation dataset consists of twenty carefully selected meme coin projects deployed on Ethereum and Binance Smart Chain. We prioritized diversity in outcomes, market sizes, and deployment timeframes to ensure the dataset represents the meme coin ecosys- tem accurately.

Tokens are categorized into three groups based on comprehensive historical evidence: **Legitimate Projects**: Established tokens with sustained community engagement and no evidence of developer exploitation, such as Dogecoin, Shiba Inu, and Pepe. These serve as positive examples of sustainable meme coin projects.

Confirmed Scams: Projects with documented rug-pull events where developers demonstrably removed liquidity or exploited contract permissions.Notable examples include Squid Game Token, which prevented users from selling while developers withdrew funds, and AnubisDAO, where developers drained the liquidity pool shortly after launch. **Moderate-Risk Projects**: Tokens showing some concerning characteristics but without confirmed fraudulent behavior.These help calibrate the system's sensitivity and reduce false positives.

Data quality and verification were prioritized throughout. Each classification was validated through multiple sources including blockchain transaction records, community reports, news articles, and developer communications.

6 How Well It Works

6.1 Performance Results

Our experimental evaluation shows that the framework achieves strong discrimination between legitimate and fraudulent meme coin projects. The Random Forest classifier achieved the highest overall performance with 89% precision, 94% recall, and an F1-score of 0.91 on the held-out test set. The ROC curve demonstrates the trade-off between true positive rate and false pos- itive rate across different classification thresholds. The area under the curve (AUC) of 0.93 significantly exceeds the random baseline of 0.50, validating the effectiveness of our approach.

We use standard classification metrics:

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN}, \quad F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{1}$$

where TP represents true positives (correctly identified scams), FP represents false positives (legitimate tokens incorrectly flagged), and FN represents false negatives (un- detected scams).

6.2 What Matters Most

Feature importance analysis reveals that liquidity lock duration, developer wallet con- centration, and ownership renouncement status contribute most strongly to prediction accuracy. This finding aligns with domain knowledge about common rug-pull mechanisms and provides confidence in the model's learned patterns.

The rule-based scoring component achieves comparable performance to the machine learning approach on this dataset, showing that the weighted linear combination of features captures the primary risk signals effectively. The ML enhancement provides marginal improvements by identifying subtle non-linear relationships, particularly in am- biguous cases where multiple moderate risk factors combine in complex ways.

6.3 Where We Made Mistakes

Error analysis of misclassified cases reveals that false positives typically involve legitimate projects with unusual tokenomics or unconventional contract implementations. False neg- atives occur primarily in sophisticated scams that temporarily maintain the appearance of legitimacy through careful manipulation of observable metrics. These findings suggest directions for future improvement including incorporation of temporal analysis and social sentiment indicators.

7 Building and Deploying It

The framework has been implemented as a modular Python application with clearly de- fined interfaces between components. This design makes it easy to integrate into existing DeFi platforms and enables independent testing and updating of individual modules.

The on-chain analysis module uses Web3.py libraries to interface with blockchain nodes. Bytecode analysis employs the Mythril symbolic execution engine for deep in- spection of unverified contracts. The off-chain analysis module interfaces with APIs from Etherscan, BscScan, DexScreener, and other data providers.

Performance optimization includes caching frequently accessed blockchain data, par- allel processing of independent analysis tasks, and incremental updates for previously analyzed tokens. These optimizations enable near real-time risk assessment suitable for integration into trading interfaces.

Deployment options include standalone web applications, browser extensions for ma- jor wallets, REST API services for third-party integration, and command-line tools for programmatic access.

8 Limitations and Future Improvements

While our framework demonstrates strong performance, several limitations warrant con- sideration. The current implementation relies on static analysis of contract code and point-in-time tokenomic snapshots. Sophisticated scammers may exploit this by initially deploying benign contracts and only enabling malicious functionality after building trust. The dataset size, while carefully curated, remains relatively small compared to the thousands of meme coins launched monthly. Expanding the training dataset would im- prove model robustness. Ongoing data collection and model retraining will be necessary to maintain effectiveness as scam tactics evolve.

Future work will incorporate temporal analysis to track how token characteristics change over time, potentially identifying gradual transitions toward rug-pull conditions. Integration of social media sentiment analysis could provide early warning signals when community perception shifts or when coordinated promotion campaigns suggest manip- ulation.

Enhanced bytecode analysis techniques including symbolic execution and formal verifi- cation could detect more sophisticated hidden vulnerabilities. Collaboration with blockchain security firms to share threat intelligence would enable faster identification of emerging attack patterns.

9 Conclusion

This paper shows that meme coins have measurable on-chain and off-chain indicators that strongly correlate with rug-pull behavior. By integrating comprehensive smart con- tract analysis, tokenomic evaluation, and machine learning classification, our Meme Coin Fundamentals Analyzer delivers an explainable and automated investor-protection mech- anism suitable for real-world DeFi

deployment.

The framework achieves high accuracy in distinguishing legitimate projects from fraudulent schemes, providing actionable risk assessments that empower investors to make informed decisions. The modular architecture ensures adaptability to evolving threats and facilitates integration into existing DeFi platforms and tools.

As the meme coin ecosystem continues to evolve, automated security analysis will become increasingly critical for protecting retail investors and maintaining trust in decen- tralized finance. Our work establishes a foundation for comprehensive rug-pull detection that can be extended and refined as new attack vectors emerge.

Acknowledgments

We thank the blockchain security community for sharing insights into emerging scam patterns and the open-source developers whose tools enabled this research.

References

1. M. Vasek and T. Moore, “Analyzing the Bitcoin Ponzi Scheme Ecosystem,” Workshop on Economics of Information Security (WEIS), 2015.
2. J. Chen, S. Li, and Y. Zhang, “Detecting Scam Tokens on Ethereum and Binance Smart Chain Using Transaction Pattern Analysis,” IEEE International Conference on Blockchain and Cryptocurrency, pp. 245–253, 2022.