Vrushali Kadam

Zeru Problem statement #2

1. Fetch Transaction History:
   - Retrieve the transaction data for each provided wallet address from compound V2 or V3 protocol.
2. Data Preparation:
   - Organize and preprocess the transaction data to create meaningful features that can reflect each wallet's risk profile.
3. Risk Scoring:
   - Develop a scoring model that assigns each wallet a risk score ranging from 0 to 1000.
   - Clearly document your feature selection, normalization method, and scoring logic.

Deliverables:

A CSV file with columns:

wallet_id	score
0xfaa0768bde629806739c3a4620656c5d26f44ef2	732

A brief explanation detailing:

- Data Collection method
- Feature selection rationale
- Scoring method
- Justification of the risk indicators used

My Approach Explanation for Wallet Risk Scoring

1. Data Collection Method

To fetch historical transactions, I used the Etherscan API. While the task asked for interaction with Compound V2 or V3, I broadened the scope to general Ethereum transactions due to limitations in open-access protocol-specific APIs.

The script collects the full transaction history for each wallet by recursively querying the Etherscan API using pagination via the startblock parameter.

2. Feature Selection Rationale

Each wallet's behavior was distilled into a rich feature set capturing both volume, diversity, and regularity of activity:

Feature	Why is it important
num_transactions	How active the wallet is – more transactions means it's used more often.
num_failed	If many transactions failed, it might be a bot or someone doing risky stuff.
total_value_eth	Shows how much ETH the wallet is moving – big numbers may mean it's a whale.
mean_value_eth, std_value_eth	Helps see if the wallet sends similar amounts or random amounts – stable vs. unstable use.
unique_counterparties	Tells how many different wallets this one talks to – could mean it's connected to DeFi or many users.
unique_function_calls, most_common_function	Shows what kind of smart contract functions the wallet uses – like supplying, borrowing, or others.
avg_time_between_txs_sec	If the time between transactions is small and repeated, it might be a bot or some fast activity.

These features are chosen to reflect user intent, scale, and stability, all of which are key indicators in lending protocol risk modeling.

3. Scoring Method
I used a multi-model risk scoring approach to ensure robustness:

MinMax Weighted Model (MM_risk_score)
A manually weighted scoring based on normalized values of key indicators like num_failed, value, etc. High weight is given to failure rates and variance, indicating instability or bot-like behavior.

Local Outlier Factor (LOF)
Detects wallets whose behavior is statistically anomalous compared to the population. Outliers often represent risk.

Isolation Forest (IF)
A tree-based model to isolate high-risk behavior, such as sharp value spikes, unusually short transaction bursts, etc.

Each of these risk models generates a score between 0–1000, and the final risk score is computed as the average of the three. This ensembling improves robustness and handles edge cases.

4. Justification of Risk Indicators

Risk indicators were chosen based on DeFi best practices and empirical patterns seen in on-chain analysis:

1. High failure rate: Indicates faulty bots, failed contract interaction.
2. Frequent small or bursty txns: Could point to automated behavior.
3. High variance in amount: Implies inconsistent financial behavior, potentially high risk.
4. Few counterparties: May signal laundering or isolated usage; very high may imply attack surfaces.
5. Outlier detection: Detects subtle patterns human intuition might miss.

5. Scalability

Uses standard libraries (pandas, sklearn, requests) and can scale horizontally across multiple addresses.

Easily extensible to protocol-specific datasets (Compound V2/V3, Aave, etc.) with ABI decoding.

Designed with batch processing in mind, with room for parallelization or off-chain enrichment.