Ethereum
Illicit Activity Mitigation

By Landon Smith, Manaswini Mishra, Ayushi Singhal,
Saddiq Jeelani, & Anamika Medhi

Foreword

Dr. Navid Sabbaghi, Program Director

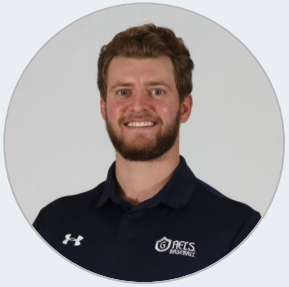
Musa Ogunyemi, CEO of AI Chain

Thank you to Dr. Navid Sabbaghi and Musa Ogunyemi! Their assistance helped this research project thrive!

Agenda

1. Team Introduction
2. Burning Platform: Why Analyze Ethereum?
3. Introduction to Ethereum
4. Contemporary Research & Our Proposed Additions
5. Result Achievement Process
6. Limitations
7. Tech Stack
8. Future Work
9. Questions

Cohort Members



Landon Smith

Ayushi Singhal

Saddiq Jeelani

Manaswini Mishra

Anamika Medhi

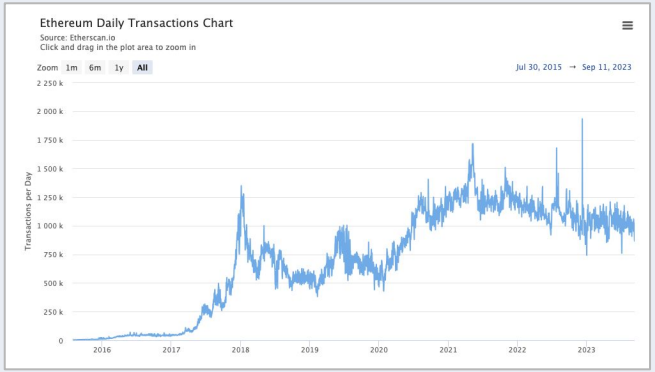
Why Analyze Ethereum?

- The rise in popularity of blockchain technology over recent years has opened the floodgates for new capital to begin flowing into blockchain protocols like Ethereum
- Spike in engagement has also given malicious actors a larger audience to prey on
- The activity present on emerging blockchain networks has now become too large for regulators of traditional financial markets to ignore
- Measures Taken to Limit Illicit Activity:
    - Implementation of KYC
    - Implementation of Sanctions
- Our cohort shares the belief that if blockchain networks like Ethereum are to go mainstream, anti-fraud measures must be in place to counteract rampant malicious activity

etherscan.io

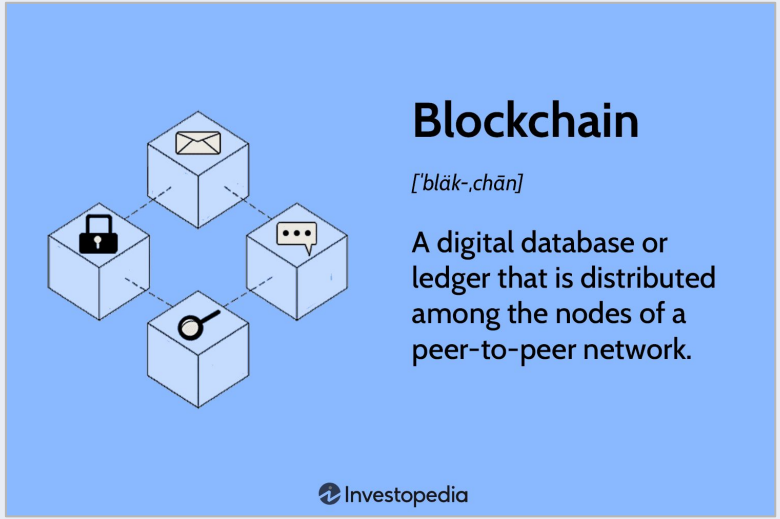
Total **7,773** nodes found

#	Countries	Last 24 Hours
1	United States	3,359 (42.65%)
2	Germany	1,612 (20.47%)
3	United Kingdom	338 (4.29%)
4	Japan	227 (2.88%)
5	Canada	213 (2.70%)
6	Singapore	202 (2.57%)
7	China	168 (2.13%)
8	Russia	142 (1.80%)
9	Lithuania	120 (1.52%)
10	Netherlands	119 (1.51%)



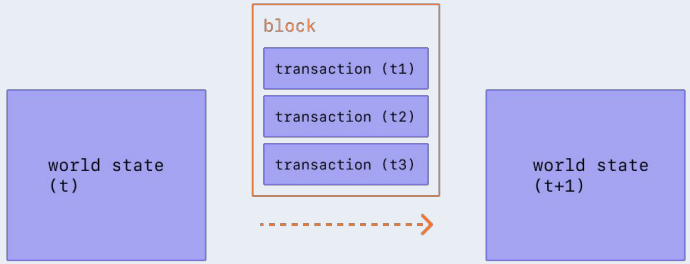
What is a Blockchain?

- A blockchain is a public database that is governed by the participating computers which host it
  - “Block” refers to a chunk of updates to the current state of the database .
  - “Chain” refers to the bonded nature of these chunks .
    - The aggregation of all blocks in the chronological chain results in the entire history of a blockchain
  - This technology originated with the conception of Bitcoin in 2009.



What is Ethereum?

- Ethereum builds on Bitcoin's innovation of blockchain technology, but layers additional technology on top
- The physical computers (nodes) that host the Ethereum network determine the state of a virtual computer called the EVM (Ethereum Virtual Machine)
- Anyone who is a participant on the network can request that this computer perform computation on their behalf
- Ethereum is the world's second largest blockchain network by overall market cap, currently standing at $200 billion



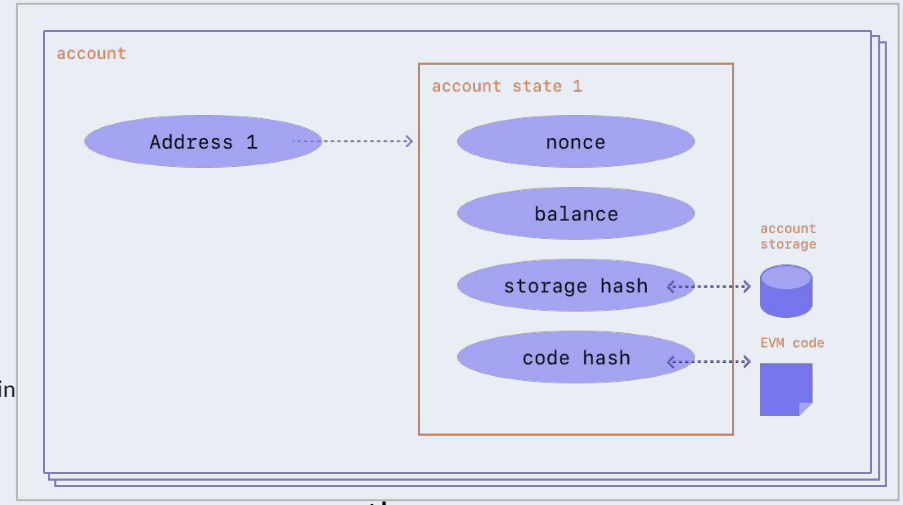
ethereum.org

What is Ether?

- The purpose of the Ether cryptocurrency is to usher in a market for computation on the EVM (Ethereum virtual Machine) and provide an economic incentive for participants to provide computational resources to the network
- The Ethereum community refers to the payments made to nodes to facilitate arbitrary computation and verify the legitimacy of transactions as “gas”

What are Addresses?

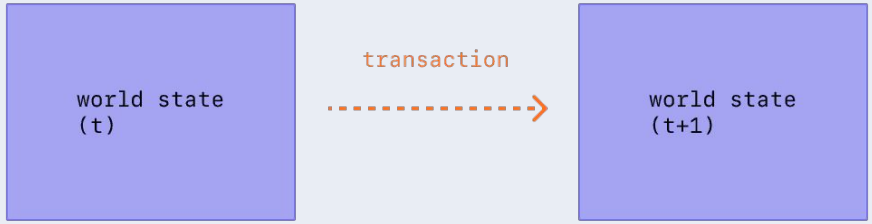
- There are two account types on Ethereum:
  - Externally Owned Accounts (EOA)
    - Creation costs nothing
    - Can initiate transactions
    - Controlled by a public & private key pair
  - Contract Accounts
    - Creation costs Ether because of network storage costs
    - Can only initiate a transaction in response to receiving a transaction
    - A contract account has a public key, but no private key
    - Contract accounts are controlled by the logic in their code
- Both account types can:
  - Receive, hold, and send Ether and other token formats supported by Ethereum
  - Interact with deployed smart contracts
- Addresses are denoted by a 42 character hexadecimal
  - 0x06012c8cf97bead5deae237070f9587f8e7a266d



ethereum.org

What are Transactions?

- Transactions change the state of the EVM(Ethereum virtual machine)
- Require inclusion in a validated block
- Require a gas fee paid to validators
- Transactions include many different fields of data
  - From
  - To
  - Value
  - Nonce
  - Input Data
  - Gas
  - etc.

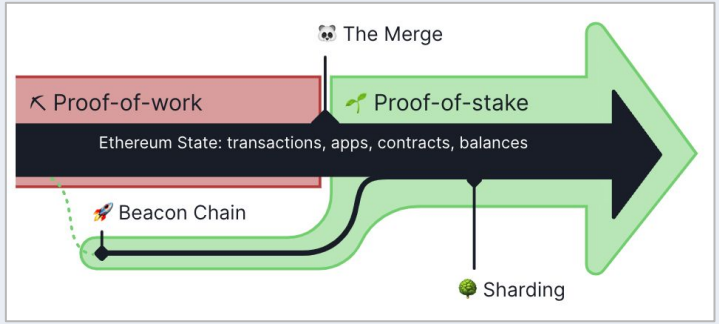


ethereum.org

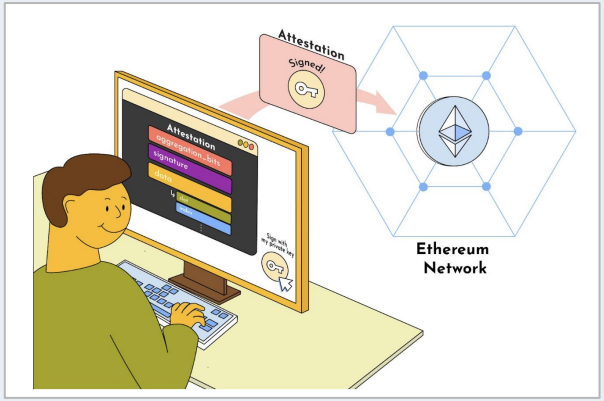
Relevant Ethereum Infrastructure

- Consensus Mechanism
  - Protocol that allows decentralized actors to coordinate and agree on the state of the blockchain
- The Merge: ETH to ETH 2.0
  - Transitioned the Ethereum Blockchain from a consensus mechanism called Proof of Work to Proof of Stake
- Proof of Stake Consensus Mechanism
  - Participants who want take part in network consensus must stake the native currency Ether as collateral
  - These validators are then responsible for attesting to newly proposed blocks, or creating them when selected to be a block proposer
  - Validators are rewarded Ether for their services

ethereum.org

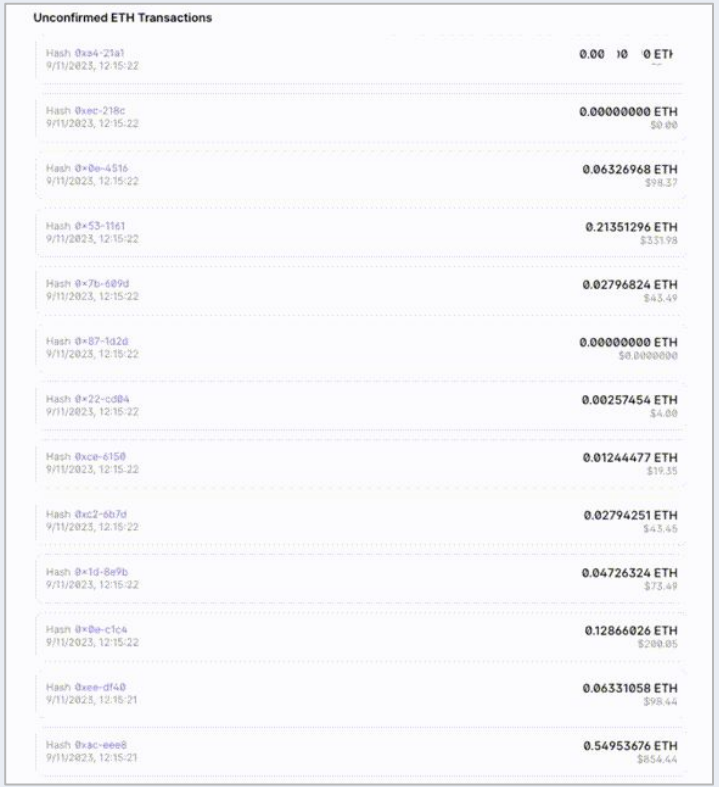


Preethi Kasireddy



Relevant Ethereum Infrastructure

- Mempool
  - The place where transactions that have not yet been added to proposed blocks are stored
  - Can be thought of as a “purgatory” for pending transactions
  - Transactions with higher gas fees are more likely to be included in proposed blocks
- MEV Boost Relays
  - Stands for maximum extractable value
  - Allows validators selected as block proposers to outsource the creation of their blocks
  - Validators can choose which relays to accept blocks from
  - While MEV Boost is not native to the Ethereum protocol, it has proliferated amongst validator nodes with usage rates at about 90%



Blockchain.com

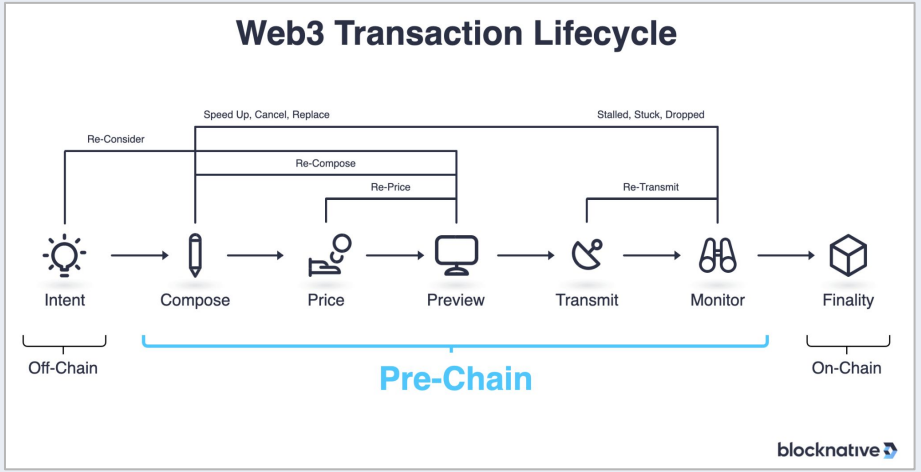
Contemporary Research

- Exploring the prior research done in the area of illicit activity detection on Ethereum revealed many studies where authors were able to create highly effective machine learning models that could distinguish between illicit and legal account activity
- We aimed to replicate these results and take them a step further in two separate areas
    - Real-Time Mempool Transaction Classification & Mitigation
    - Multi-categorization Classification

Study Author	Model Type	Accuracy	F1-Score
Farrugia	XGBoost	96.3%	96%
Pahuja & Kamal	LightGBM	99.2%	99%
Alarab & Prakoonwit	XGBoost	98.91%	97.60%

Contemporary Research Additions: Real-Time Mempool Transaction Monitoring

- Infrastructure to deploy a similar model into production on the real-time transaction activity of accounts on Ethereum
    - Utilize model in a preventative way
    - Disallow illicit activity to occur on chain in the first place
- Our cohort foresees a future where an anti-fraud system could be integrated into the Ethereum transaction lifecycle
    - Enforced by regulatory agencies on nodes within certain jurisdictions
    - Adopted emergently by decentralized network of nodes
- Rejection of illicit transactions is a form of censorship
- Is there precedent for validator nodes accepting transaction censorship?



blocknative.com

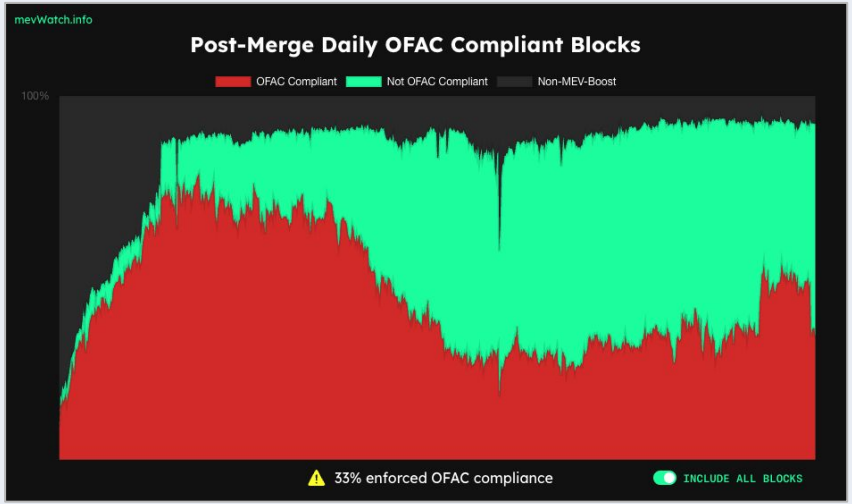
Censorship Precedent: Tornado Cash

- The Office of Foreign Assets Control (OFAC) is the financial intelligence and enforcement agency of the US Treasury Department
- Tornado Cash is a smart-contract that operates on the Ethereum blockchain which functions as a cryptocurrency mixer service to help users ensure privacy
- The addition Tornado Cash to the SDN list signaled a stark pivot from OFAC's prior enforcement actions
    - OFAC typically designates people or organizations as being illegal for US citizens to transact with
    - Tornado Cash marked the first time a decentralized entity was added to the list
    - The usage of code was blacklisted
- Despite 80% of activity on Tornado Cash being legitimate according to Elliptic, the smart contract was blanket sanctioned for allowing the Lazarus Group, who are a North Korean state-sponsored hacking group, to launder over $455 million

blockworks.com

Sanctions Compliance: MEV Watch

- How did validator nodes respond to the new OFAC sanctions on Tornado Cash?
- Certain nodes have run MEV Boost relays which censor Tornado Cash transactions, while others utilize relays which do not enforce such restrictions
- The chart tracks the percentage of blocks built by OFAC compliant relays
- While Ethereum is a decentralized blockchain, there are external centralizing forces such as MEV Boost through which the effects of regulation can be felt

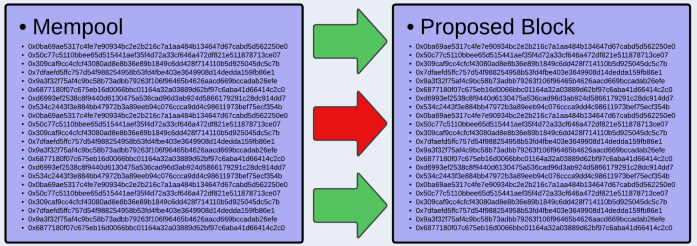


mevwatch.info

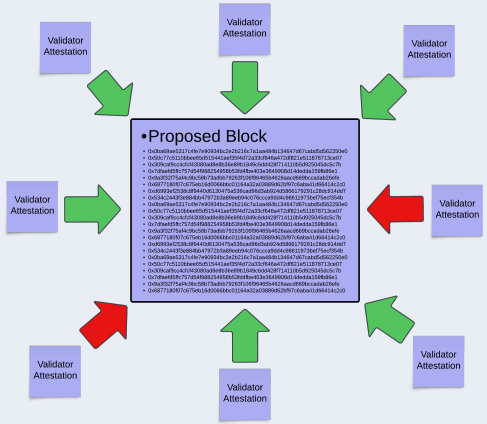
Contemporary Research Additions: Real-Time Mempool Transaction Monitoring

- Two main methods of deployment
  - Method #1: Deploy model on real-time mempool transactions, helping block proposers to avoid selecting transactions that are potentially fraudulent in their proposals
  - Method #2: Deploy model on the entirety of transactions within freshly proposed blocks, helping validators to avoid attesting to blocks that contain fraudulent transactions
- Widespread utilization of the model by nodes creates game theoretical decisions to be made by ecosystem actors
  - Block proposers receive a reward based on their block being successfully finalized
  - Block proposers will not submit blocks that have fraudulent transactions in them due to the risk of other validators declining to attest to the said block

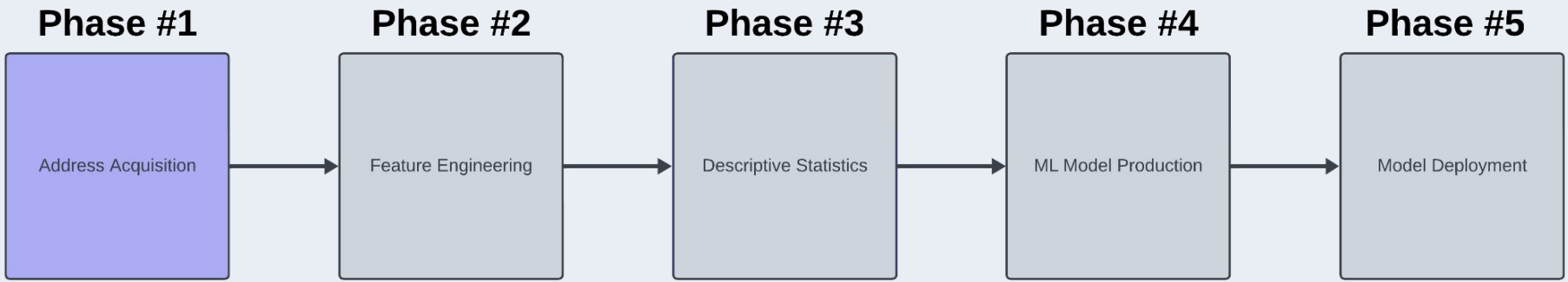
Method #1



Method #2



Solution Roadmap



Acquisition of Illicit Accounts

- Accounts flagged by the Ethereum community as fraudulent were considered to be illicit
- Gathered illicit accounts from datasets used in other academic studies pertaining to the field
    - Farrugia Dataset
    - Aliyev Dataset
    - Escobero Dataset
- Enriched our dataset with additional illicit accounts reported on various fraud tracking websites
    - OFAC
    - EtherScamDB
    - Chainabuse
    - Etherscan
- There was overlap from each source, but after removing any duplicate accounts we were left with 11,378 illicit accounts in our final dataset
- Illicit addresses were flagged as 1

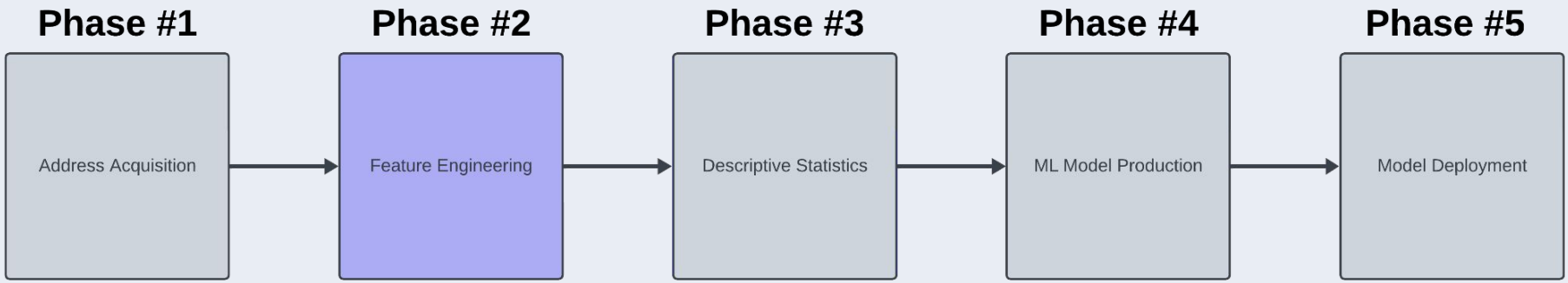
Acquisition of Legal Accounts

- We have created two separate lists of addresses deemed as legal that adhere to different criteria
- Random Sampling of Addresses
    - Iterated through block range 17999000 to 18000000, isolating every account that was involved in a transaction within each block
    - The number of accounts obtained was far greater than the number of illicit accounts that we had acquired previously, so we shuffled the addresses and randomly selected 15,000 addresses to be used as legal accounts
- Addresses with Coinbase Interactions
    - Coinbase enforces KYC on its exchange, may give us better representation of legal activity
    - Iterated through block range 17995000 to 18000000 and isolated addresses that interacted with Coinbase addresses tagged by Etherscan
- We cross-checked both address lists with our illicit address list to ensure that there was no overlap
- Legal addresses were flagged as 0

VS.

coinbase

Solution Roadmap

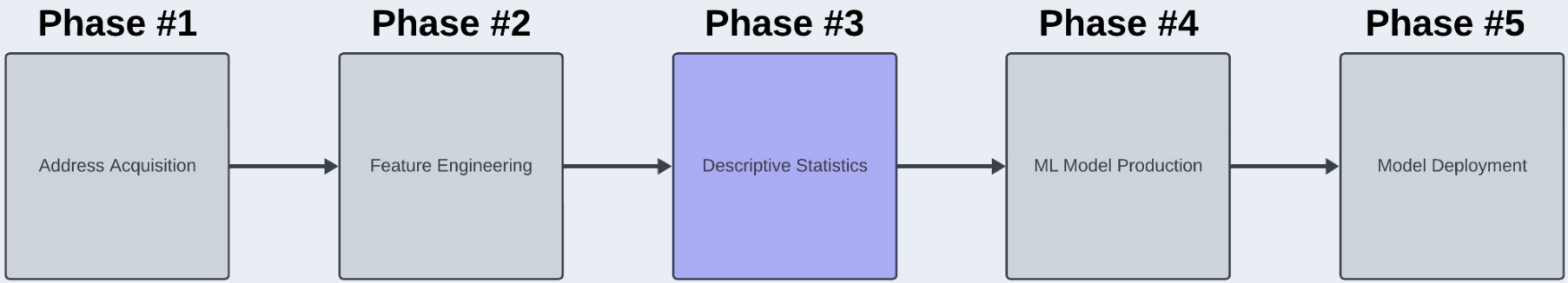


Feature Engineering

- Once we had accumulated all of our addresses, we utilized the API of blockchain explorer Etherscan to gather aggregate transaction information on each address to generate features
- We requested the following information from the Etherscan API for each address:
    - Current ETH Balance
    - All Normal Transactions
    - All ERC20 Transactions
- We then utilized custom functions to extract more precise features from the above data retrieved
- In total, we generated 22 features on each address in our datasets
- Features were generated up to block 18,000,000

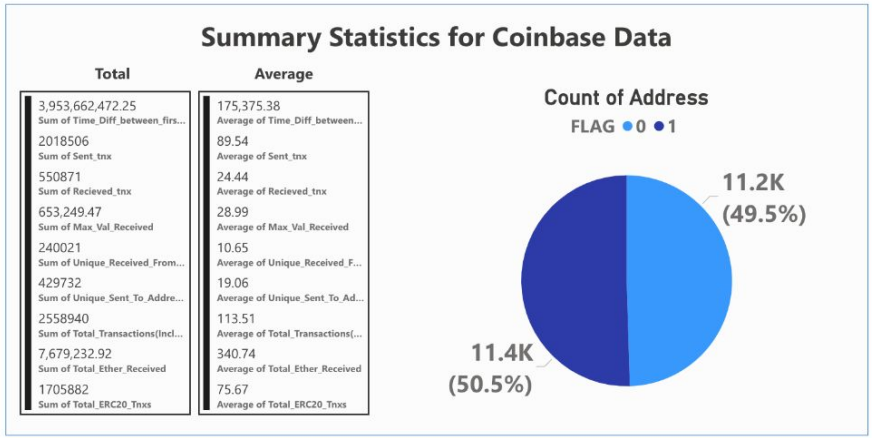
Feature	Description	Data Type
Sent_tnx	The # of transactions sent by the address	Integer
Recieved_tnx	The # of transactions received by the address	Integer
Total_Transactions	The total number of transactions the address has either sent or received	Integer
Total_Ether_Balance	The current balance of the address in Ether (Block 18,000,000)	Float
Max_Value_Received	The maximum amount of Ether received in a single transaction	Float
Min_Value_Received	The minimum amount of Ether received in a single transaction	Float
Total_Ether_Received	The total amount of Ether received by the address across all address transactions	Float
Max_Value_Sent	The maximum amount of Ether sent in a single transaction	Float
Min_Value_Sent	The minimum amount of Ether sent in a single transaction	Float
Total_Ether_Sent	The total amount of Ether sent by the address across all address transactions	Float
Avg_Value_Received	The average Ether value received across all address transactions	Float
Avg_Value_Sent	The average Ether value sent across all address transactions	Float
Time_Delta_First_Last	The difference in time between an addresses first and last transaction in minutes	Float
Avg_min_between_received_tnx	The average number of minutes between an addresses received transactions	Float
Avg_min_between_sent_tnx	The average number of minutes between an addresses sent transactions	Float
Unique_Received_From_Addresses	The number of unique addresses the address received Ether from	Integer
Unique_Sent_To_Addresses	The number of unique addresses the address sent Ether to	Integer
Total_ERC20_Tnxs	The total number of ERC20 transactions the address has either sent or received	Integer
ERC20_Avg_Time_Between_Rec_Tnx	The average number of minutes between an addresses received ERC20 transactions	Float
ERC20_Avg_Time_Between_Sent_Tnx	The average number of minutes between an addresses sent ERC20 transactions	Float
ERC20_Uniq_Rec_Addr	The number of unique addresses the address received ERC20 tokens from	Integer
ERC20_Uniq_Sent_Addr	The number of unique addresses the address sent ERC20 tokens to	Integer

Solution Roadmap

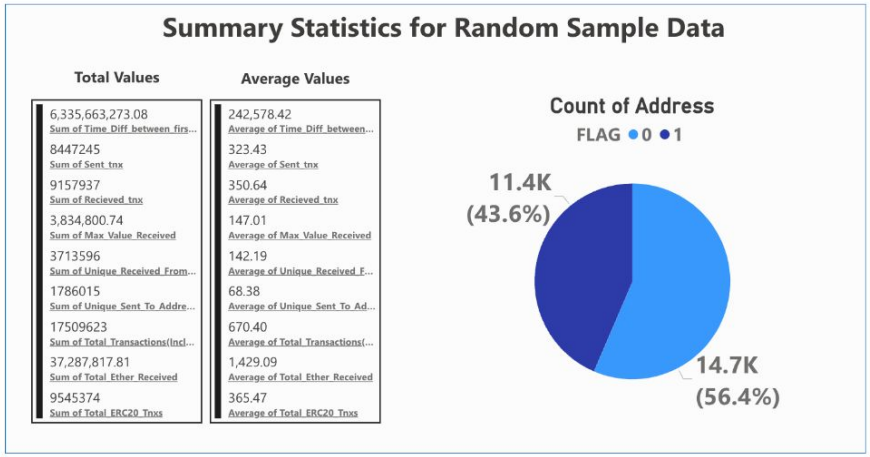


Descriptive Statistics on Datasets

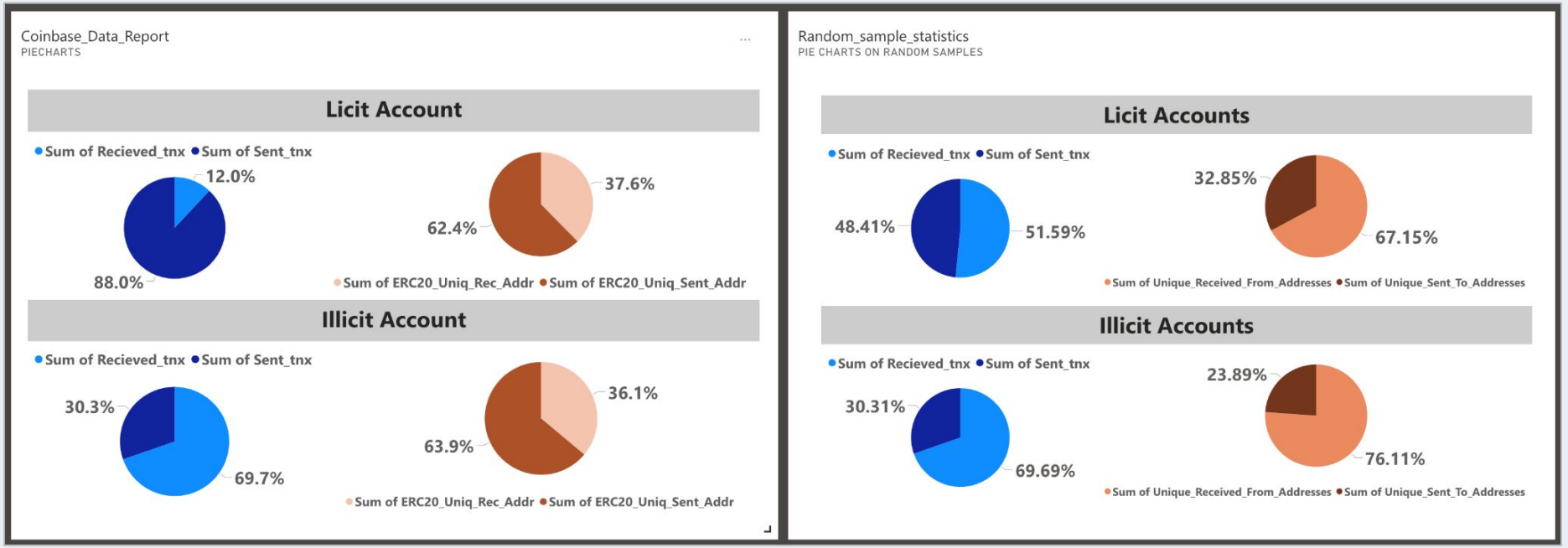
Coinbase_Data_Report
SUMMARY STATS COINBASE



Random_sample_statistics
SUMMARY STATITICS



Descriptive Statistics on Datasets



Average Value Statistics of Coinbase data

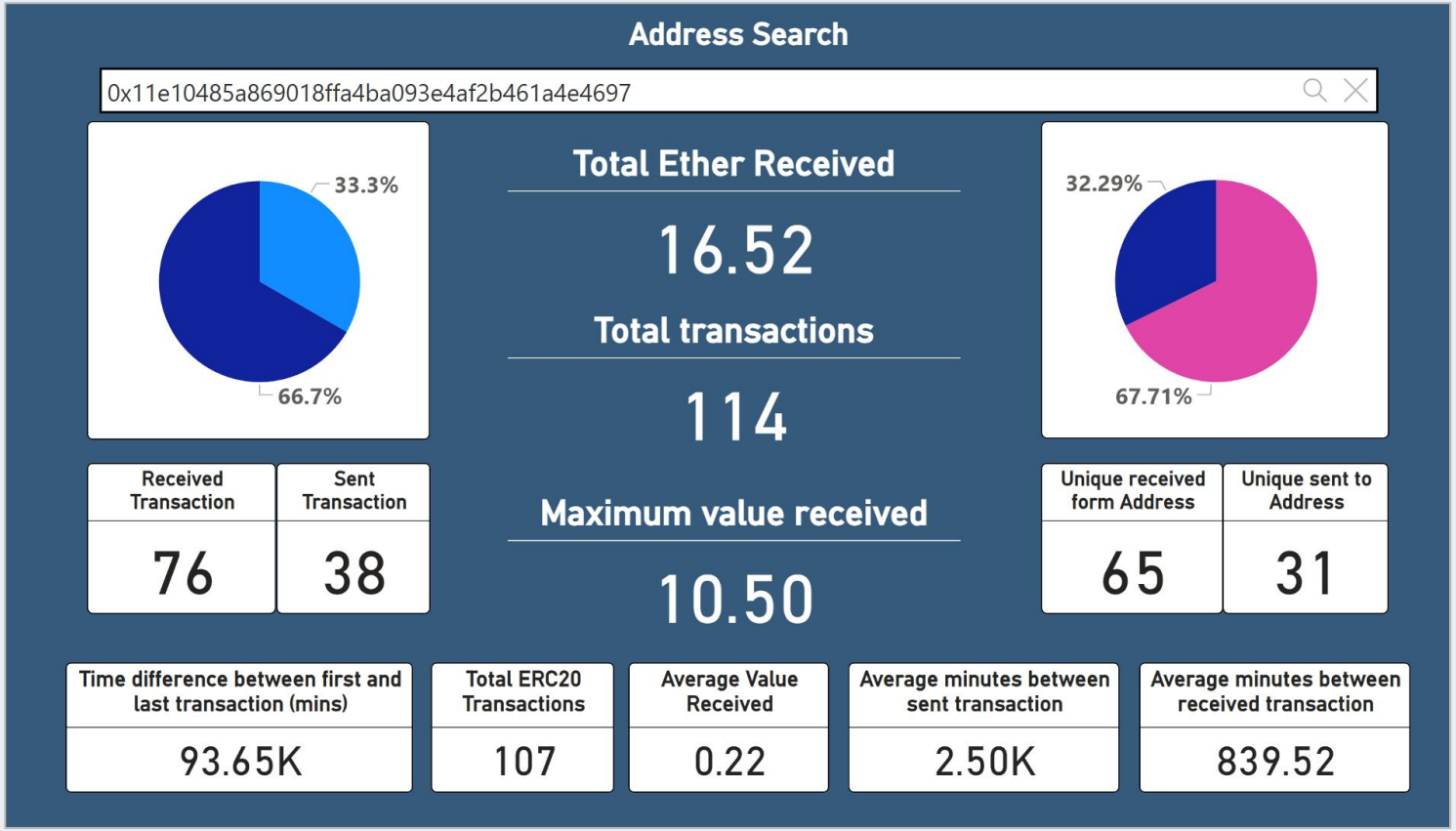
Total Ether received	ILLICIT	LICIT	Average of Time_Diff_(Mins) by FLAG	Average of Max_Val_Received by FLAG	Average of Recieved_tnx by FLAG	Average of Avg_min_between_received_tnx by FLAG	Average of Avg_min_between_sent_tnx by FLAG
>500	152	41	1.2M 0.5M	1.1K 3.1K	478 498	16K 84K	3K 16K
500-400	33	14	0.92M 0.62M	49 200	220 116	15K 2K	1K 22K
400 -300	45	46	1.1M 0.4M	68 136	276 210	19K 68K	0.00M 0.13M
300 -200	77	83	1.2M 0.3M	31 112	185 116	15K 9K	2.0K 7.3K
200 -100	180	147	1.0M 0.4M	24 53	129 128	17K 15K	4K 29K
100 -1	3323	3416	0.54M 0.21M	3.2 7.0	40 36	31K 11K	15K 27K
1 -0	6523	7232	69K 30K	0.097 0.042	1.8 2.8	14K 3K	9.2K 6.7K

Random_sample_statistics
AVERAGE STATS ON RANDOM SAMPLE

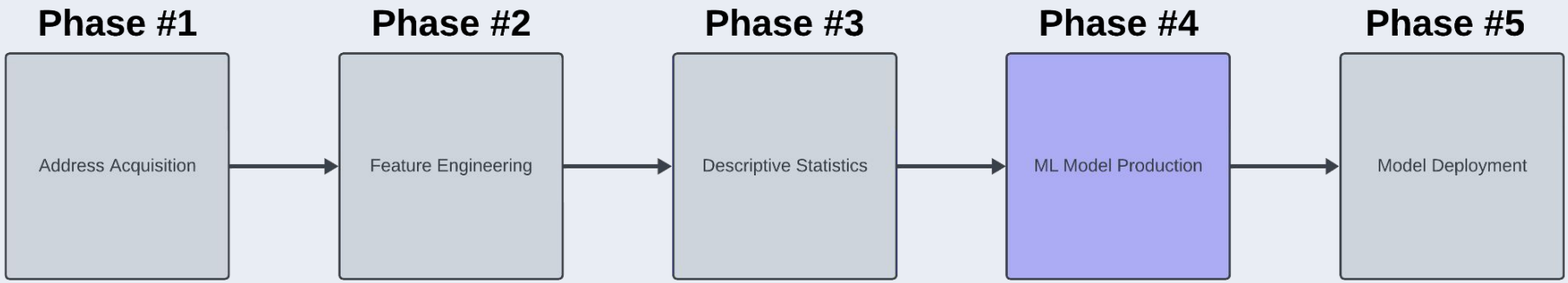
Average Statistics on Random Samples Data

Total Ether Received	ILLICIT	LICIT	Average of Time_Diff_between_first_and_last_(Mins) by FL...	Average of Max_Value_Received by FLAG	Average of Recieved_tnx by FLAG	Average of Avg_min_between_received_tnx by FLAG	Average of Avg_min_between_sent_tnx by FLAG
>500	152	633	0.53M / 0.51M	5.2K / 3.1K	3.6K / 0.5K	5K / 84K	1K / 16K
500-400	33	66	0.81M / 0.30M	89 / 200	1.1K / 0.1K	14K / 2K	5K / 22K
400-300	45	74	0.77M / 0.41M	60 / 136	1.6K / 0.3K	1.1M / 3.0M	0.01M / 0.13M
300-200	77	154	0.86M / 0.30M	40 / 112	810 / 116	14K / 9K	2.3K / 7.3K
200-100	180	348	0.84M / 0.38M	25 / 53	458 / 128	16K / 15K	4K / 29K
100-1	3323	6220	0.48M / 0.21M	3.6 / 7.0	151 / 36	31K / 11K	12K / 27K
1-0	6523	5997	0.17M / 0.03M	0.15 / 0.04	82 / 3	39K / 3K	0.15bn / 0.04bn

Dashboard on Datasets



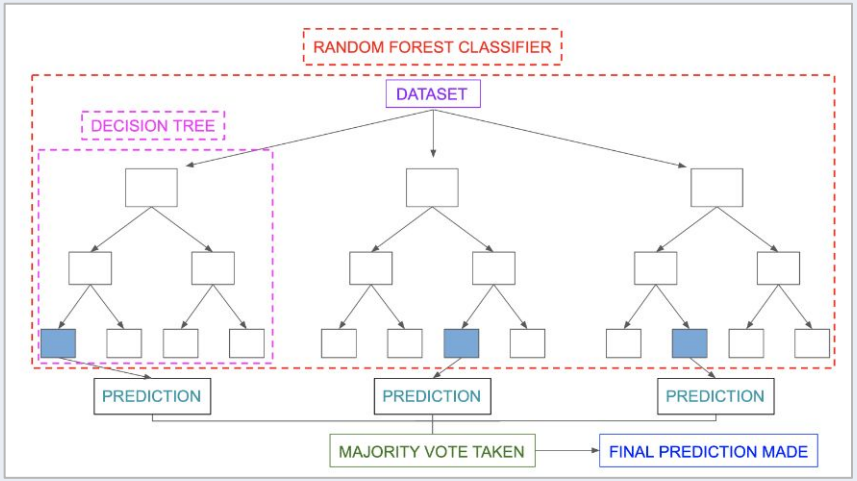
Solution Roadmap



Machine Learning Model Production: Supervised and Unsupervised Learning Methods

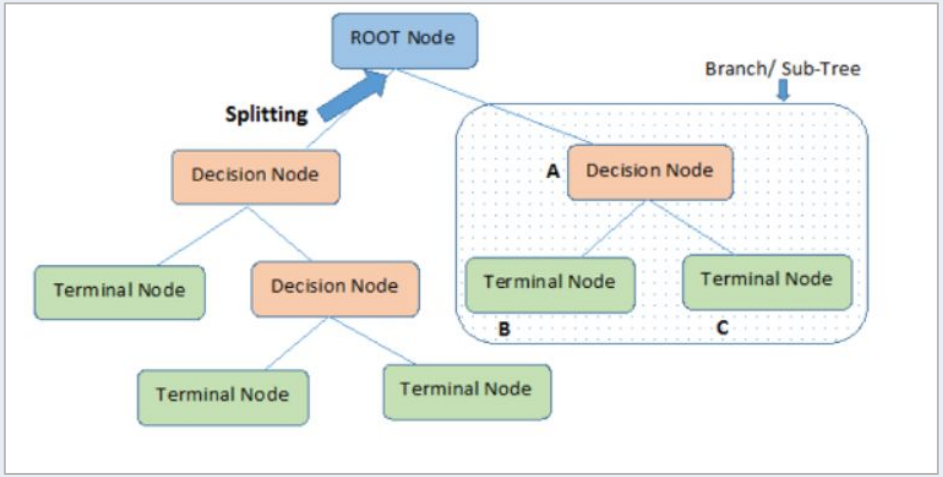
	Supervised Learning	Unsupervised Learning
Methods	Random Forest, Logistic, Decision Tree, ADABoost, Gradient Boosting, XGBoost, Balanced Bagging Classifier	Isolation Forest
Pros	Better Accuracy; Needs less computation resources and time	No need for labels; New fraud patterns can be learnt
Cons	Need training for new fraud patterns	Need more computation resources
Use case	Detects fraudulent	Detects Funnel health for future frauds

Classification Model - Random Forest



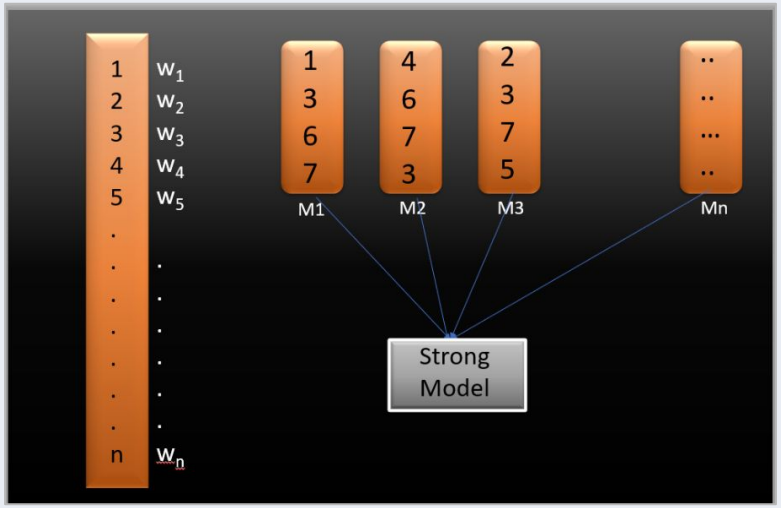
- Random forest is an ensemble algorithm comprising multiple decision trees, and it uses bagging (bootstrap aggregating) to improve machine learning algorithm accuracy.
- Random forest predicts outcomes by averaging the predictions from its decision trees, and increasing the number of trees enhances prediction precision.
- A random forest ensemble combines multiple decision trees trained on randomly selected subsets of data to make predictions through majority voting.

Classification & Regression Model - Decision Trees



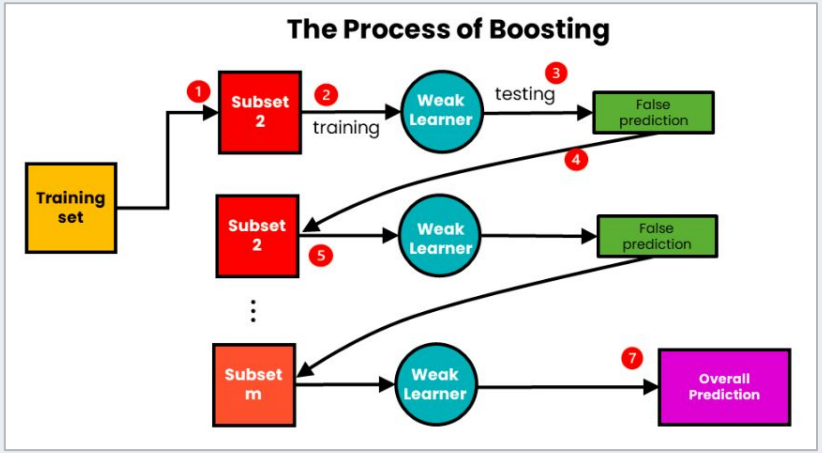
- Decision Trees are the supervised learning models used for both classification and regression problems.
- A decision tree is a hierarchical model for decision support that represents choices, outcomes, and probabilities. It uses conditional statements, suitable for classification and regression. It has a tree structure with root, branches, internal nodes, and leaf nodes.
- Decision trees are versatile tools for classification and regression. They use a tree-like structure to make predictions, starting at a root and ending with leaf decisions.

Classification Model - Gradient Boost Classifier



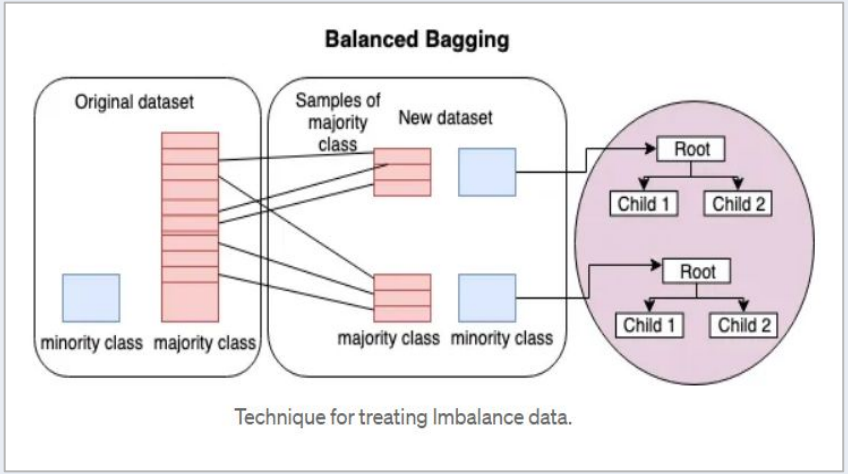
- Boosting is an ensemble learning technique that builds models sequentially, aiming to reduce errors by focusing on the mistakes of previous models. It creates new models based on the errors or residuals of the previous ones.
- The goal is to minimize a loss function by adding weak learners through gradient descent. Different loss functions, like MSE for regression and log-likelihood for classification, are used accordingly.

Classification Model - XGBoost



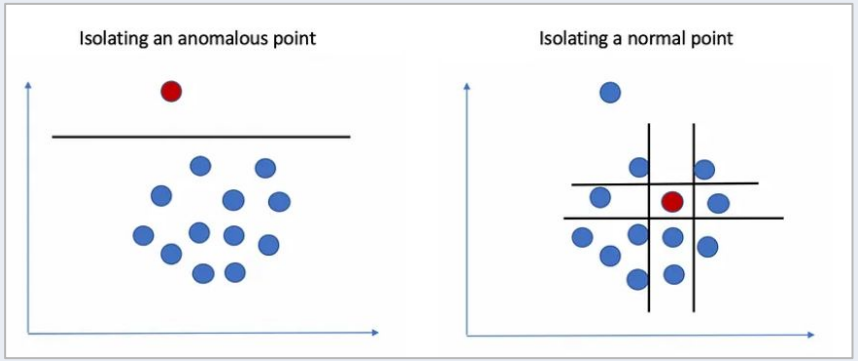
- XGBoost is a robust machine-learning algorithm that can help you understand your data and make better decisions.
- XGBoost is an implementation of gradient-boosting decision trees. It has been used by data scientists and researchers worldwide to optimize their machine-learning models.

Balanced Bagging Classifier



- The Balanced Bagging Classifier is an ensemble method that trains multiple classifiers on different subsets of data and combines their predictions.
- It tackles imbalanced datasets by applying random undersampling to balance class distributions within each subset.
- By mitigating bias towards the majority class, it improves overall model performance, especially on minority class predictions.
- It is particularly useful for tasks like fraud detection and medical diagnosis, where class imbalance can impact accuracy.

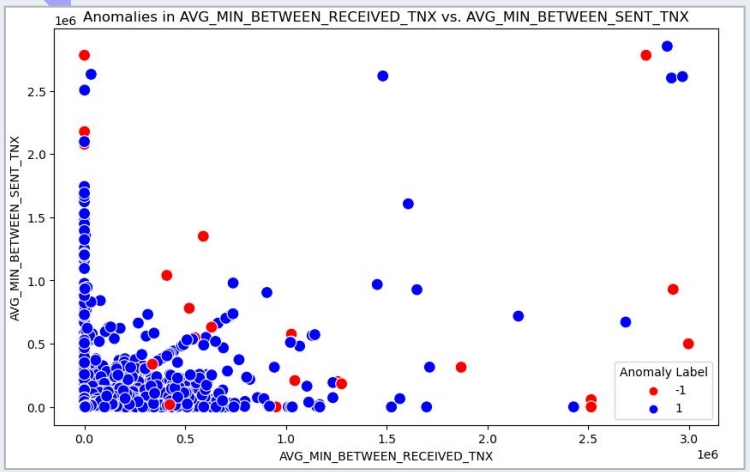
Anomaly Detection - Isolation Forest



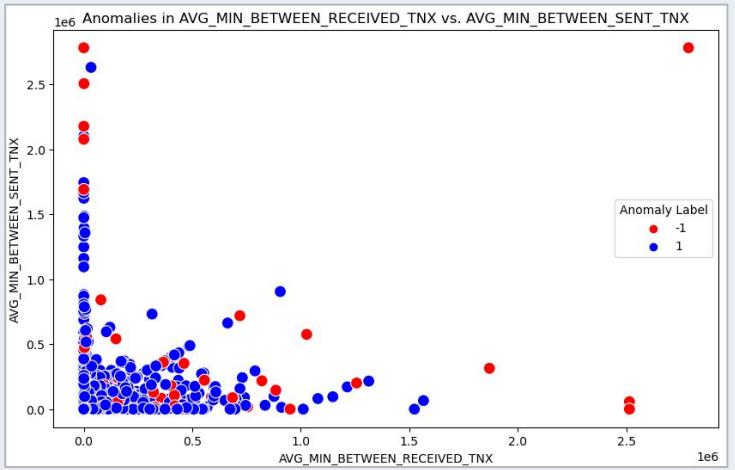
- It's an unsupervised technique for anomaly detection that processes randomly sub-sampled data using a tree structure and is one of the ensemble techniques.
- The tree structure is built based on randomly selected features from the data
- Samples that traverse deeper into the tree are less likely to be anomalies because they require more splits to isolate.
- Conversely, samples ending up in shorter branches are more likely to be anomalies because they were easier for the tree to separate from the rest of the data.
- Isolation Forest uses the simplicity of isolating anomalies in shorter branches to identify them efficiently.

Anomaly Detection - Isolation Forest

Random Sampling



Coinbase



- Here the negative values displayed in red represents the anomalies that could be seen in both random sampling and coinbase datasets
- This can help in analyzing the impacts the data points can make in the funnel

Framework for Choosing Anomaly Detection Solution

Parameter 1: Availability of labelling

- Blockchain being novel technology will witness evolving nature of illicit transactions
- Data labels will be required to train supervised models in order to keep up with new patterns of illicit transactions
- If data labels are not available, an unsupervised model is better for deployment due to cold start problem else supervised models will train better to detect illicit transactions

Parameter 2: Availability of computation resources

- Due to heavy traffic on Ethereum blockchain and its immutable nature, computation resources to process transactions is a driving factor in deciding model deployment
- Supervised models will use less computation resources while unsupervised will use high computation resources

Parameter 3: Desired Accuracy and Precision

- Supervised models have high accuracy of more than 94% while Isolation forest i.e. unsupervised model cannot make explicit predictions for the classes but can be visually interpreted based on the thresholds that we keep

Model Statistics for Test Dataset

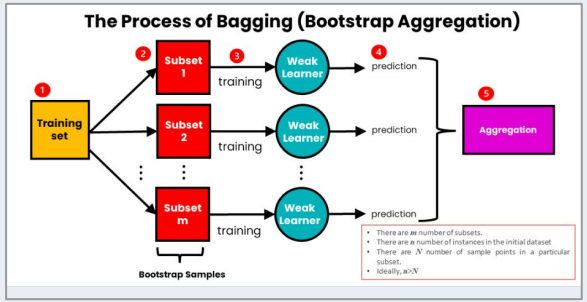
Coinbase Dataset

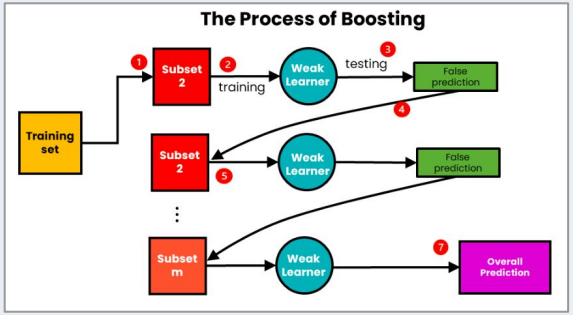
	Model	Accuracy	Precision	Recall
0	Random Forest	0.972501	0.977501	0.967761
1	DNN Model	0.709196	0.646285	0.935522
2	Decision Trees	0.949586	0.943658	0.957210
3	Gradient Boost	0.974571	0.978441	0.970985
4	XGBoost	0.974571	0.978441	0.970985
5	ADABoost	0.939237	0.908466	0.846087
6	Logistic Regression	0.690568	0.797315	0.345650
7	Balanced Bagging Classifier	0.966750	0.977464	0.955360

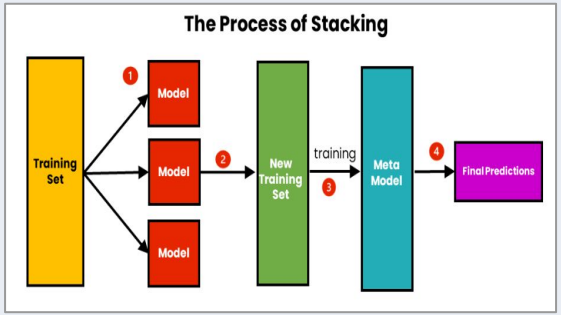
Random Sampling

	Model	Accuracy	Precision	Recall
0	Random Forest	0.941041	0.957270	0.906023
1	DNN Model	0.708525	0.610293	0.928135
2	Decision Trees	0.909265	0.900882	0.891184
3	Gradient Boosting	0.945253	0.954381	0.919116
4	XGBoosting	0.945253	0.954381	0.919116
5	ADA Boost	0.895100	0.908466	0.846087
6	Logistic Regression	0.674451	0.797315	0.345650
7	Balanced Bagging Classifier	0.941768	0.953067	0.929105

Ensembling Modelling



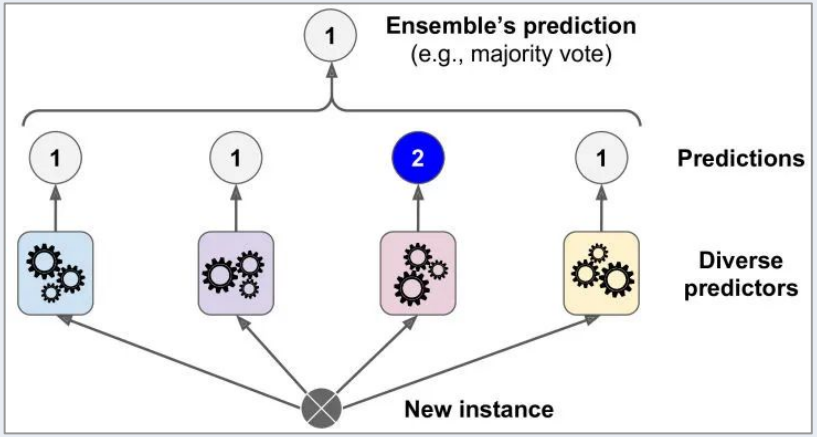




- Ensemble learning combines multiple machine learning models to improve prediction accuracy by mitigating the limitations of individual models, which can be either high bias or high variance, resulting in poor performance.
- The goal of ensemble learning is to balance the bias-variance trade-off by reducing bias or variance, depending on the characteristics of the weak learners, ultimately creating a strong learner that can make accurate predictions.

Ensemble Modelling - Selection of the technique

	Bagging	Boosting	Stacking
Purpose	Reduce Variance	Reduce Bias	Improve Accuracy
Base Learner Types	Homogeneous	Homogeneous	Heterogeneous
Base Learner Training	Parallel	Sequential	Meta Model
Aggregation	Max Voting, Averaging	Weighted Averaging	Weighted Averaging



- Ensemble, in essence, involves the amalgamation of multiple models, where a group of models is employed to make predictions as opposed to relying on a single model.
- Considering the various ensembling techniques we used the bagging technique through Voting Classifier.
- Here we choose multiple base classifiers along with one balanced classifier model that are trained independently on the same training datasets for two different ones.
- Each base classifier provides its own predictions and the final predictions are made by taking the soft voting i.e. weighted average of class probabilities.
- This helps to have have models trained simultaneously that expedites the processing.

Ensembling Results

Coinbase Dataset

```
Model Score: 0.972
Precision is: 0.98
Recall is: 0.964
F1 score is: 0.972
```

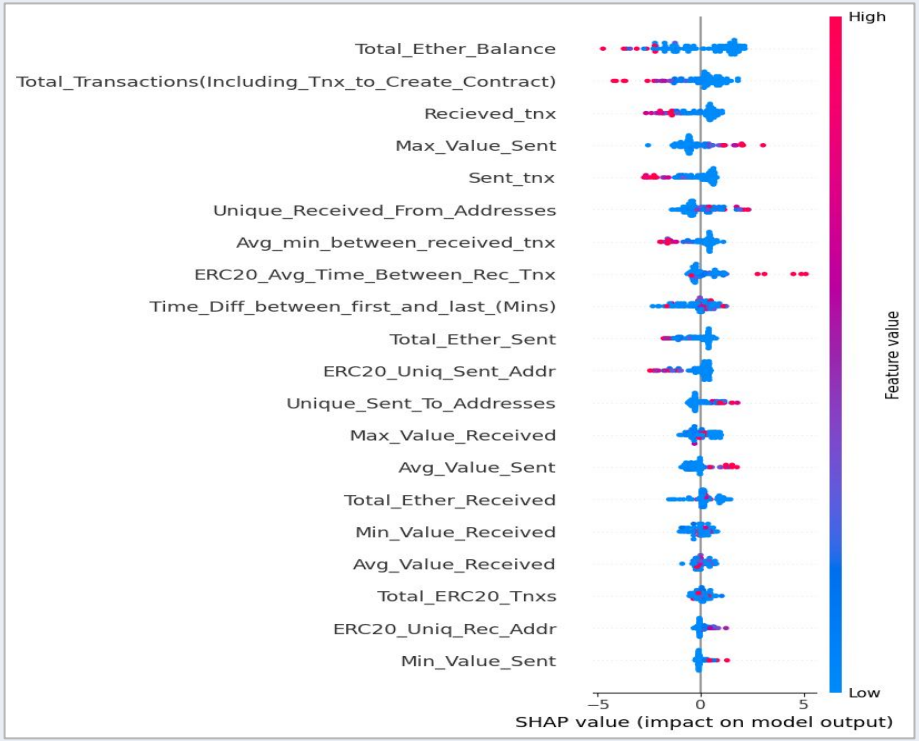
	Model	Accuracy	Precision	Recall
0	Random Forest	0.972501	0.977501	0.967761
1	DNN Model	0.709196	0.646285	0.935522
2	Decision Trees	0.949586	0.943658	0.957210
3	Gradient Boost	0.974571	0.978441	0.970985
4	XGBoost	0.974571	0.978441	0.970985
5	ADABoost	0.939237	0.908466	0.846087
6	Logistic Regression	0.690568	0.797315	0.345650
7	Balanced Bagging Classifier	0.966750	0.977464	0.955360

Random Sampling

```
Model Score: 0.947
Precision is: 0.956
Recall is: 0.937
F1 score is: 0.947
```

	Model	Accuracy	Precision	Recall
0	Random Forest	0.941041	0.957270	0.906023
1	DNN Model	0.708525	0.610293	0.928135
2	Decision Trees	0.909265	0.900882	0.891184
3	Gradient Boosting	0.945253	0.954381	0.919116
4	XGBoosting	0.945253	0.954381	0.919116
5	ADA Boost	0.895100	0.908466	0.846087
6	Logistic Regression	0.674451	0.797315	0.345650
7	Balanced Bagging Classifier	0.941768	0.953067	0.929105

Feature Importance Using SHAP Value



SHAP (Shapley Additive exPlanations) values are a mathematical framework used in machine learning to explain the contribution of individual features to a model's prediction, helping to understand the importance of each feature in the overall outcome.

Feature Importance using ELI5 value

"Eli5" is a casual acronym for "Explain Like I'm 5." It's a request for a simple, easy-to-understand explanation of a complex topic, often used when seeking simplified explanations for complex concepts or ideas.

Weight	Feature
0.0151 ± 0.0047	Unique_Received_From_Addresses
0.0148 ± 0.0047	Time_Diff_between_first_and_last_(Mins)
0.0078 ± 0.0016	Avg_received_time
0.0057 ± 0.0032	Avg_sent_time
0.0048 ± 0.0063	total_ether_balance
0.0030 ± 0.0039	total_ether_received
0.0030 ± 0.0024	Total_ERC20_tnxs
0.0028 ± 0.0028	ERC20_max_val_rec
0.0024 ± 0.0026	ERC20_avg_val_rec
0.0023 ± 0.0032	min_val_sent
0.0020 ± 0.0050	ERC20_min_val_rec
0.0020 ± 0.0033	ERC20_total_Ether_received
0.0016 ± 0.0014	ERC20_uniq_rec_contract_addr
0.0014 ± 0.0027	ERC20_uniq_rec_addr
0.0013 ± 0.0006	ERC20_uniq_sent_addr
0.0011 ± 0.0021	Sent_tnx
0.0009 ± 0.0050	avg_val_received
0.0009 ± 0.0006	ERC20_avg_val_sent
0.0009 ± 0.0011	ERC20_uniq_sent_token_name
0.0009 ± 0.0017	ERC20_uniq_rec_token_name
… 25 more …	

Weight	Feature
0.0765 ± 0.0084	Total_Ether_Balance
0.0392 ± 0.0038	Unique_Received_From_Addresses
0.0313 ± 0.0060	Max_Value_Sent
0.0273 ± 0.0049	Time_Diff_between_first_and_last_(Mins)
0.0235 ± 0.0052	Total_Transactions(Including_Tnx_to_Create_Contract)
0.0203 ± 0.0035	Avg_min_between_received_tnx
0.0178 ± 0.0022	Total_Ether_Received
0.0158 ± 0.0013	Max_Value_Received
0.0152 ± 0.0005	ERC20_Avg_Time_Between_Rec_Tnx
0.0131 ± 0.0041	Sent_tnx
0.0126 ± 0.0033	Unique_Sent_To_Addresses
0.0122 ± 0.0028	Recieved_tnx
0.0087 ± 0.0003	Total_Ether_Sent
0.0060 ± 0.0017	Avg_Value_Sent
0.0057 ± 0.0029	ERC20_Uniq_Sent_Addr
0.0049 ± 0.0015	Total_ERC20_Tnxs
0.0042 ± 0.0037	Min_Value_Received
0.0031 ± 0.0019	ERC20_Uniq_Rec_Addr
0.0019 ± 0.0019	Avg_Value_Received
0.0011 ± 0.0009	Min_Value_Sent
… 2 more …	

Randomly Sampled Dataset: SHAP & ELI5 Value Comparison

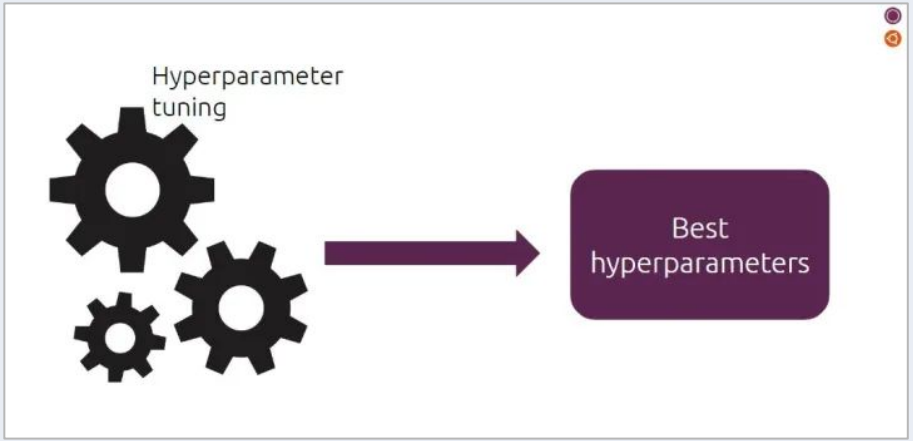
	Feature	SHAP Value	ELI5 Score
10	avg_val_received	1.644044	0.018221
6	Unique_Received_From_Addresses	1.359009	0.056085
2	Time_Diff_between_first_and_last_(Mins)	1.255815	0.034021
21	total_ether_balance	0.794783	0.019359
19	total_ether_received	0.755829	0.006690
34	ERC20_min_val_rec	0.744273	0.016512
17	total_transactions_(including_tnx_to_create_co...	0.715248	0.018363
0	Avg_sent_time	0.564767	0.009537
8	min_value_received	0.508687	0.004413
1	Avg_received_time	0.469679	0.006263
11	min_val_sent	0.337245	0.005836
3	Sent_tnx	0.334496	0.003559
4	Received_Tnx	0.304310	0.000569
22	Total_ERC20_tnxs	0.301049	0.005267
9	max_value_received	0.269156	0.005979
18	total_Ether_sent	0.252686	0.000854
13	avg_val_sent	0.200310	0.002847
35	ERC20_max_val_rec	0.163555	0.002135
12	max_val_sent	0.155261	-0.000285
23	ERC20_total_Ether_received	0.114046	0.003132
7	Unique_Sent_To_Addresses	0.103628	-0.000569
37	ERC20_min_val_sent	0.087114	0.001423

Coinbase Dataset:
SHAP & ELI5 Value Comparison

	Feature	SHAP Value	ELI5 Score
2	Total_Ether_Balance	1.457897	0.076531
7	Total_Transactions(Including_Tnx_to_Create_Con...	0.867066	0.023507
9	Max_Value_Sent	0.799176	0.031317
1	Recieved_tnx	0.710725	0.012175
0	Sent_tnx	0.692422	0.013093
18	Unique_Received_From_Addresses	0.681634	0.039242
14	Avg_min_between_received_tnx	0.565244	0.020291
11	Total_Ether_Sent	0.543370	0.008691
6	Time_Diff_between_first_and_last_(Mins)	0.519221	0.027259
17	ERC20_Avg_Time_Between_Rec_Tnx	0.503329	0.015237
20	ERC20_Uniq_Sent_Addr	0.478868	0.005743
12	Avg_Value_Sent	0.472235	0.005972
19	Unique_Sent_To_Addresses	0.462041	0.012596
3	Max_Value_Received	0.438108	0.015773
5	Total_Ether_Received	0.425417	0.017802
4	Min_Value_Received	0.358318	0.004211
8	Avg_Value_Received	0.265145	0.001876
15	Total_ERC20_Tnxs	0.215810	0.004900

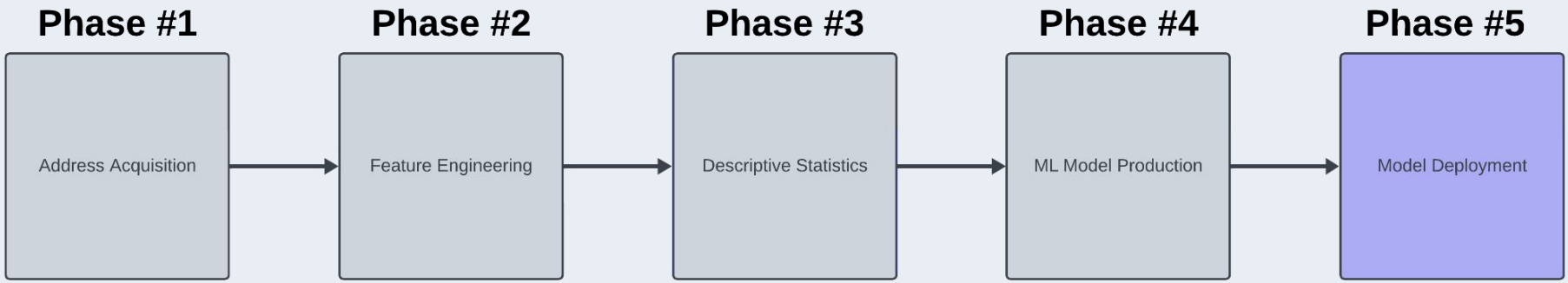
Hyperparameter Tuning on Models

- 22 Categories
- Hyperparameter tuning done on several models including Random Forest, Gradient Boost, XGBoost, Decision Tree, AdaBoost, Logistic regression models.
- We have used grid search method for tuning the hyperparameter.
- We have got Higher precision and recalls after tuning the model.
- Model was Successful and we have got better accuracy as well.



Models	Best Hyper Parameters/Grid Search Parameters
Random Forest Classifier	Number of estimators: [100, 200, 300],max_depth: [None, 5, 10, 20], min_samples_split: [2, 5, 10],min_samples_leaf: [1, 2, 4]
Decision Tree Classifier	max_depth: [None, 5, 10, 20],min_samples_split: [2, 5, 10], min_samples_leaf: [1, 2, 4],
XGBoost	Number of estimators-[100,200,300], max_depth: [3, 4, 5], learning_rate': [0.01, 0.1, 0.2]
DNN Keras Classifier	Number of neurons-128, number of layers-4,activation function-RELU.Output-Sigmoid
Gradient Boost	Number of estimators: [100, 200, 300],learning_rate: [0.01, 0.1, 0.2], max_depth: [3, 4, 5],
Ada Boost	Number of estimators: [50, 100, 200],learning_rate: [0.01, 0.1, 0.2]

Solution Roadmap



Overview of Proposed Solution: Real-Time Transaction Monitoring

- How will we feed our new ML models transactions happening in real time?
- We must first generate features for each of the accounts associated with each transaction in order to feed them to our ML models
    - The available time frame we have to make decisions is 10 seconds without creating an impediment to the speed of the blockchain
        - This is the average pace at which Ethereum blocks are validated
    - The feature engineering process isn't fast enough to be utilized for a real-time use case
        - 3 Etherscan API calls
        - Calculation time of subsequent features

Overview of Proposed Solution: Real-Time Transaction Monitoring

- Our solution to this problem comes with the implementation of a Google Cloud SQL database which would store addresses and their corresponding model classifications
    - Before engineering features from scratch:
        - If the address is in the database, we take the result associated with the model's prior decision
        - If the address isn't in our database, we feature engineer from scratch and store the results to the database for later use on future encounters
- Limitations
    - Assumption of continued behavior
    - Database would need to accumulate a large amount of addresses before being viable for real-time usage
        - Could be run asynchronously until enough addresses are gathered, then deployed

Cloud SQL

Database Schema

Whitelist Table	
Tag	*VARCHAR(32)*
Address	*VARCHAR(42)*
Flag	*INTEGER*

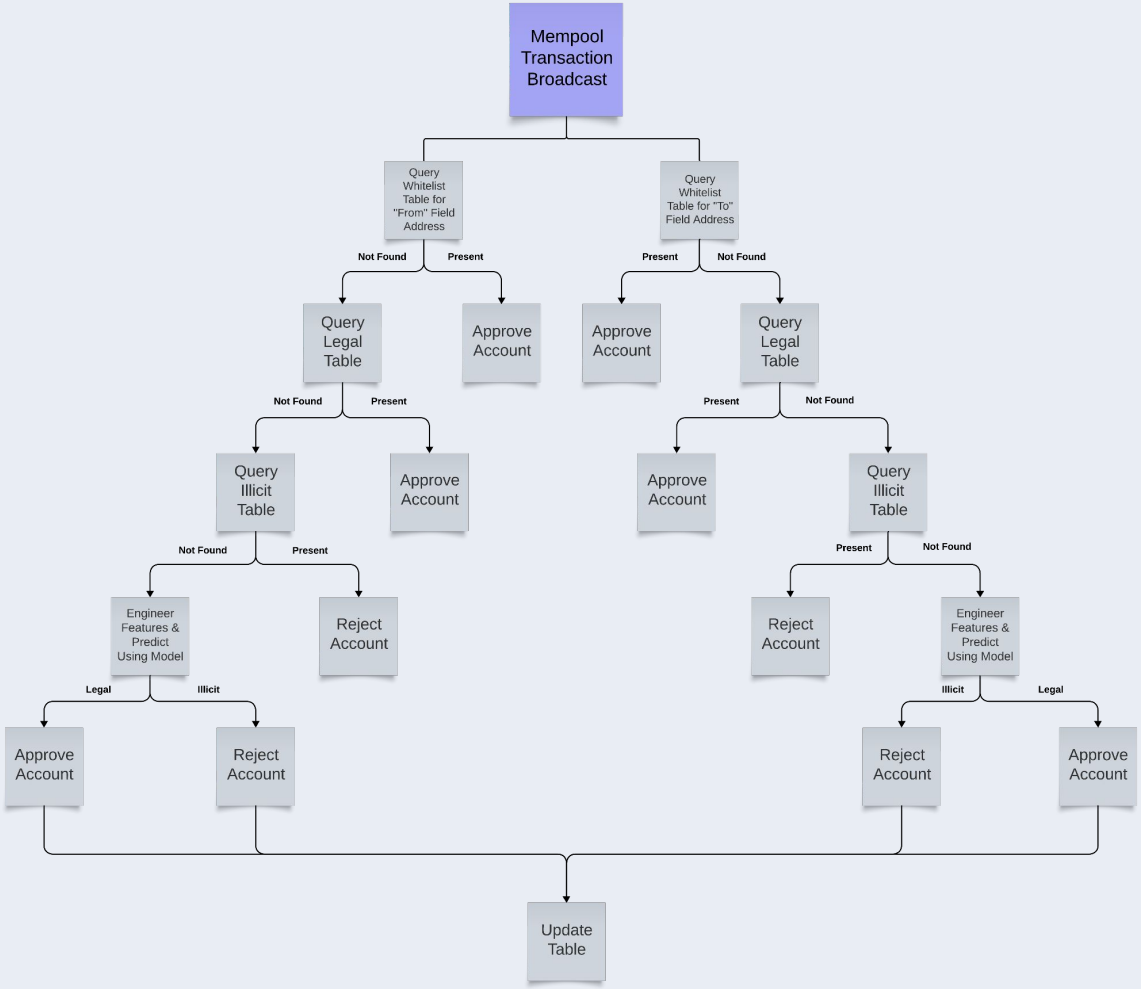
- Whitelist Table
  - Holds major crypto exchanges so that the model does not have to process them when deployed
  - Flag will be 0

Legal Address Table	
Address	*VARCHAR(42)*
Flag	*INTEGER*
Timestamp	*TIMESTAMP*

- Legal Address Table
  - Holds all addresses deemed to be legal by the model
  - Flag will be 0

Illicit Address Table	
Address	*VARCHAR(42)*
Flag	*INTEGER*
Timestamp	*TIMESTAMP*

- Illicit Addresses Table
  - Holds all addresses deemed to be illicit by the model
  - Flag will be 1


Mempool Transaction Broadcast
Query Whitelist Table for "From" Field Address
Query Whitelist Table for "To" Field Address
Not Found
Present
Present
Not Found
Query Legal Table
Approve Account
Approve Account
Query Legal Table
Not Found
Present
Present
Not Found
Query Illicit Table
Approve Account
Approve Account
Query Illicit Table
Not Found
Present
Present
Not Found
Engineer Features & Predict Using Model
Reject Account
Reject Account
Engineer Features & Predict Using Model
Legal
Illicit
Illicit
Legal
Approve Account
Reject Account
Reject Account
Approve Account
Update Table

Mempool Performance:
Coinbase Model Vs. Random Sampling Model

- When implementing our models on real-time mempool transactions, we obtained unexpected results
- We noticed a disparity between our Coinbase account-based model and our randomly sampled account-based model
    - Coinbase Model Illicit: 35%
    - Randomly Sampled Model Illicit: 25%
- Our models capture a what our cohort initially believed to be a much higher rate of illicit activity than we believe is actually present on the Ethereum blockchain
- This warranted further investigation

Coinbase Model Illicit Activity

35%

Random Sampling Model Illicit Activity

25%

External Ethereum Illicit Activity Estimations

- We observed that there are a wide range of estimates from multiple external sources
- Chainalysis pegs illicit transactions at 0.25% of overall crypto traffic
- The Financial Action Task Force (FATF) has estimated that 0.6% to 9.9% of all Bitcoin transactions are illicit
- Solidus Labs found that 8% of all Ethereum ERC20 tokens were illicit, with 12% of all BEP-20 tokens on Binance chain being illicit
- While these percentages describe different blockchains, they give us a ballpark estimate of what our results should be
- Our model's predictions are 2-3x higher than the highest external estimates we could find

Chainalysis

FATF

SOLIDUS LABS

Real-Time Mempool Classification: Performance Reflection & Limitations

- After assessing the performance of our models on real-time mempool data, we have determined that the novel methodologies proposed in aforementioned studies do not identify illicit activity as effectively as once believed
- When juxtaposed the estimations of external sources with the results of our model, it is apparent that our model has a false positive rate that we cannot accept for real-time deployment
- Our cohort believes that this issue could stem from many different causes and that there is more work to be done at all stages to refine the approach

Study Author	Model Type	Accuracy	F1-Score
Farrugia	XGBoost	96.3%	96%
Pahuja & Kamal	LightGBM	99.2%	99%
Alarab & Prakoonwit	XGBoost	98.91%	97.60%

Ensemble Model	Accuracy	F1-Score
Coinbase	97.2%	97.2%
Random Sample	94.7%	94.7%

Limitations:
Real-Time Mempool Classification

- Different transaction types & token standards
  - Txs between EOA addresses
  - Txs to contract addresses
  - Txs involving Ether
  - Fungible Tokens (ERC-20)
  - Non-Fungible Tokens (ERC-721)
  - Etc.
- Account-based ML model speed issue
  - The process of engineering abstract features for each address that is implicated in a transaction is cumbersome to our classification speed, requiring the inclusion of a database
- Assumptions made in current deployment strategy
  - We currently make the assumption that addresses deemed legal will continue to act legally into the future
  - We don't have regular updates to legal addresses built into our database infrastructure
- Etherscan's API only provides a maximum of 10,000 transactions per query
  - For the vast majority of addresses this is manageable, but this limit makes evaluating the transaction activity of highly active addresses challenging
- Flagging Criteria
  - No solid criteria to define what a legal addresses activity entails
  - Changing our definition of what a legal address is changed our results dramatically

Contemporary Research Additions: Multicategorical Classification

Transition from binary to categorical:

- Expanded from simple "Illicit or Not?", or binary classifications, to "Which Illicit Type?", or multicategorical one

Fraudulent Scam Categories:

  - Phishing
  - Impersonation
  - Rug Pull Scam
  - … And 14 more
- **Importance**:
-
  - Better threat understanding
  - Enhanced safety measures

Acquisition of Illicit Accounts By Category

Extract Illicit Ethereum Addresses

16 Categories, total ~4,000 addresses extracted

- Website extracted from:
  - Chainabuse.com
- **Methodology:**
  - Manual login
  - Navigate pages individually using:
    - XPath
    - CSS Selector
    - Element
  - Randomized delays
  - Scroll for content
  - Extract addresses
  - Use logging() instead of print() to see errors if any
- **Output:**
  - Create CSV file & save data there
  - Data: Address, Page, Scam_Type
  -
- IMPERSONATION dataset had 221 pages to scrape

Scraping Tools Utilized: Multi-Categorical Classification

- **Tools Utilized**
- **Selenium on Chrome browser**
  - Multiple User Agents
  - Varying times user agents are used
  - Emulate multiple and diverse browsers
- **Data Validation:**
  - Check for duplicates at multiple points
  - Remove from CSV
- **Roadblocks**
  - Run on VM for more processing power
  - Required stronger error handling
    - We later used logging() to debug automated scraping

Overview: Multi-Categorical Classification

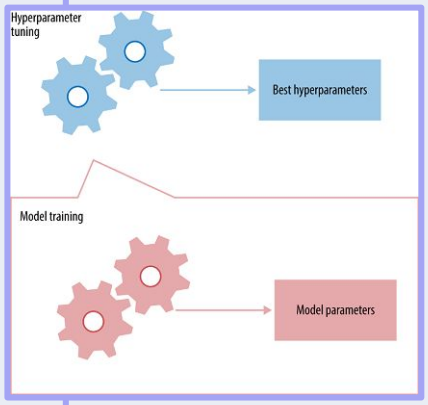
- **Dataset Details:**
  - 17 categories
  - Balanced: 1000 phishing & 1000 non-phishing samples
- **Modeling Techniques Used:**
  - AdaBoost
  - Gradient Boosting
  - Decision Tree
  - Random Forest
- **Hyperparameter Tuning:**
  - Utilizing GridSearchCV for more accurate precision
  - Manually tuned n_estimators, learning rate, max_depth
- **Ensemble Method:**
  - Voting Classifier
  - Stacking Classifier
  - Logistic Regression with models as a final_estimator
- **Performance:**
  - Highest score achieved using Stacking Classifier: 77.9%
  - Highest score after tuning base_learners: 78.1%
    - This was done by increasing max_depth

Reports by Category

Phishing Scam	23k
Impersonation Scam	3,340
Other Blackmail Scam	189
Hack - Other	186
Contract Exploit Scam	186
Fake Returns Scam	165
NFT Airdrop Scam	111
Pigbutchering Scam	108

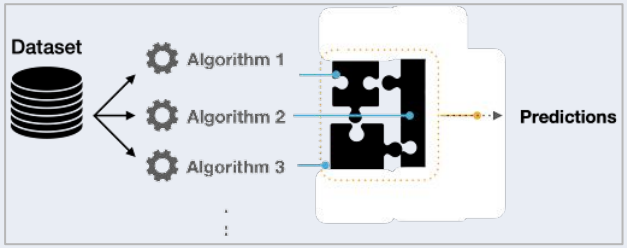
Hyperparameter Tuning: Multi-Categorical Classification

- **Model Ensemble:** RandomForest, GradientBoosting, AdaBoost, Decision Tree.
- **Key Hyperparameters:**
  - RF: n_estimators=100, max_depth=15
  - GB & AB: n_estimators=100, learning_rate=0.1
- **Classification Focus:** Initial binary focus - Phishing vs. Non-Phishing due to 27:1 data imbalance.
- **Model's Best Result:** 78.19% Accuracy
- **Balanced Detection:** Precision & Recall reveal efficient tuning across classes.
- **Future Aim:** Refine 'Non-Phishing' categorization for granular scam identification
- **End Goal:** Input CSV, output potential illicit activity types.



Ensemble Modeling: Multi-Categorical Classification

- Model Techniques: Voting & Stacking for optimal decision-making.
- Ensemble Components: RandomForest, GradientBoosting, AdaBoost, Decision Tree.
- **Voting Classifier Performance:**
  - Accuracy: 74.18%
  - Precision (Phishing): 0.81
  - Recall (Phishing): 0.65
- **Stacking Classifier Performance:**
  - Accuracy: 77.94%
  - Precision (Phishing): 0.82
  - Recall (Phishing): 0.73
- Optimal Configuration: Utilized best models from individual grid searches.
- **Benefit:** Averages model biases, boosts overall predictability.
- **Result:** Precision categorization of phishing vs. 'Non-Phishing' classes.

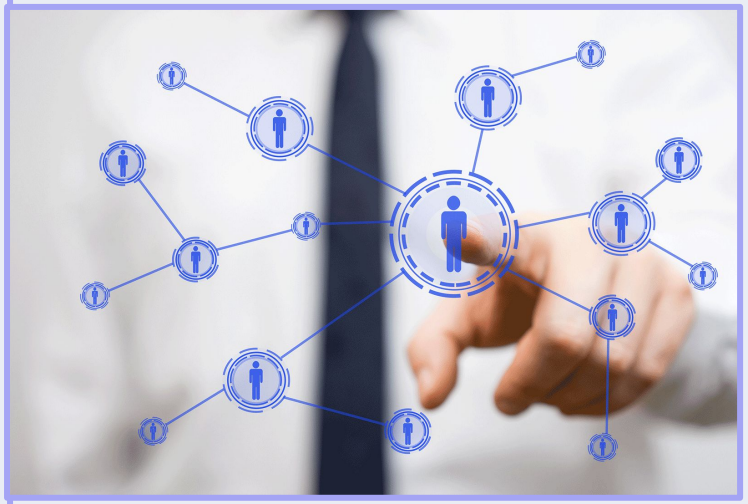


Further Classification: Overview

- **Background**
  - Initial Setup: Initially classified Ethereum addresses as licit or illicit using a FLAG column (0 for licit, 1 for illicit).
- **Limited Categorization**
  - Scam Types: Originally could only distinguish between PHISHING and NON-PHISHING/OTHER categories.
- **Objective**
  - Enhancement: Aim to extend predictive capabilities for a more detailed classification of scams.
- **Final Goal**
  - New Column: Introduce a "Scam_Type" column to predict specific scam types including IMPERSONATION, enhancing data granularity.

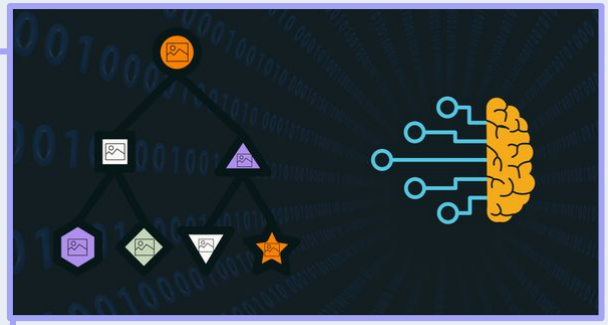
Further Classification: Data Integration and Enhancement

- **Datasets:**
  - **Dataset 1:** Contained detailed scam categories aligned with Ethereum addresses.
  - **Dataset 2 (Test data):** Had Ethereum addresses marked as licit or illicit but lacked detailed scam categorizations.
- **Integration:**
  - Scam Types Unification: Lesser frequent scam categories were aggregated under "OTHER_SCAM" to maintain a balanced dataset.
  - Incorporation of IMPERSONATION: Identified and separated all rows featuring the "IMPERSONATION" scam type to facilitate a three-category prediction system.



Further Classification: Predictive Modeling

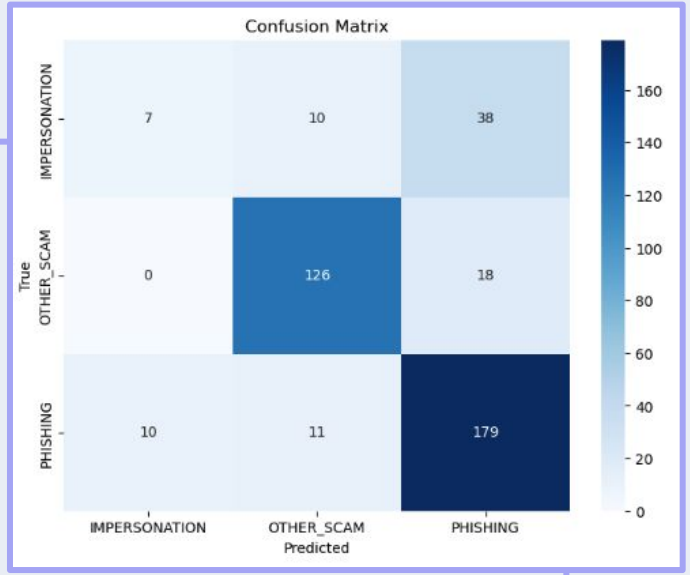
- **Model Enhancement:**
  - Detailed Classification: Using the enhanced dataset to train a machine learning model capable of predicting the specific scam types, diving deeper than just the licit-illicit or PHISHING/OTHER binary distinction.
  - Testing: Utilizing a portion of the newly created dataset to test the predictive accuracy of the enhanced model.
- **Future Prospects:**
  - Integration with Previous Models: The improved model to work in tandem with previous models to first predict the licit or illicit nature of an account and subsequently determine the specific scam type of illicit addresses.
  - Deployment: Leverage joblib for deploying the model, facilitating easy reuse for future datasets.



Further Classification: Model Performance Metrics

IMPERSONATION, PHISHING, OTHER_SCAM (excluding LICIT and NA categories)

- **Confusion Matrix Metrics:**
  - Key Scam Types:
    - IMPERSONATION:
      - Precision: 0.41 (accurately predicted 41%)
      - Recall: 0.13
    - OTHER_SCAM:
      - Precision: 0.86 (accurately predicted 86%)
      - Recall: 0.88
    - PHISHING:
      - Precision: 0.76
      - Recall: 0.90 (actual cases captured: 90%)
- **Model Accuracy:**
  - Overall accuracy: 78%
- **Averages** (weighted and unweighted across all):
  - Overall Avg: 0.68
  - Weighted Avg: 0.75

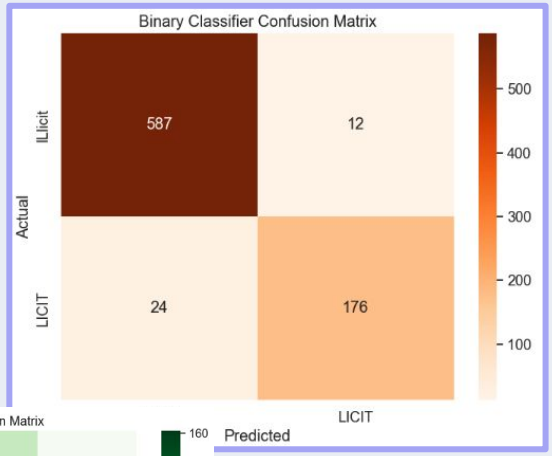


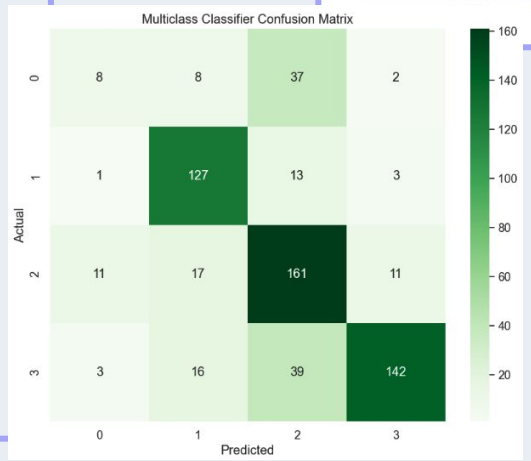
Actual \ Predicted	Impersonation	Other Scam	Phishing
Impersonation	7	10	38
Other Scam	0	126	18
Phishing	10	11	179

Further Classification: Hierarchical Model

Binary Classification First, Multiclass Classification Second

- **Objective:** Classify addresses into scam types; LICIT, ILlicit, and various forms of scams.
- **Data Handling:** Read data from 'merged_dataset.csv' and handle missing values.
- **Binary Classification:** Distinguish between LICIT and ILlicit addresses.
- **Multiclass Classification:** Classify into specific scam types (excluding LICIT and Unknown addresses).
- **Random Forest:** Chosen for its versatility and ability to handle complex data.
- **Model Evaluation:** Use accuracy and classification report to understand model performance.
- **Predict Unknowns:** Use the trained model to categorize the 1000 uncategorized addresses.





Results Compared: Multi-Categorical Classification

Binary Classification Accuracy: Achieved 95.49%.

Top Performers in Binary Classification: ILlicit category with 97% f1-score.

Multiclass Classification Accuracy: Achieved 73.12%.

Varied Performance: Best for 'Unknown' with 79% f1-score, lowest for 'IMPERSONATION' with 21% f1-score.

Model Strength: Random Forest provided a balanced classification for multiple scam types.

Uncategorized Addresses: Successfully predicted scam types for 1000 unknown addresses.

Next Steps: Focus on improving 'IMPERSONATION' and other low-performing categories.

Binary Model Validation Accuracy: 0.9549436795994993
Binary Model Classification Report:

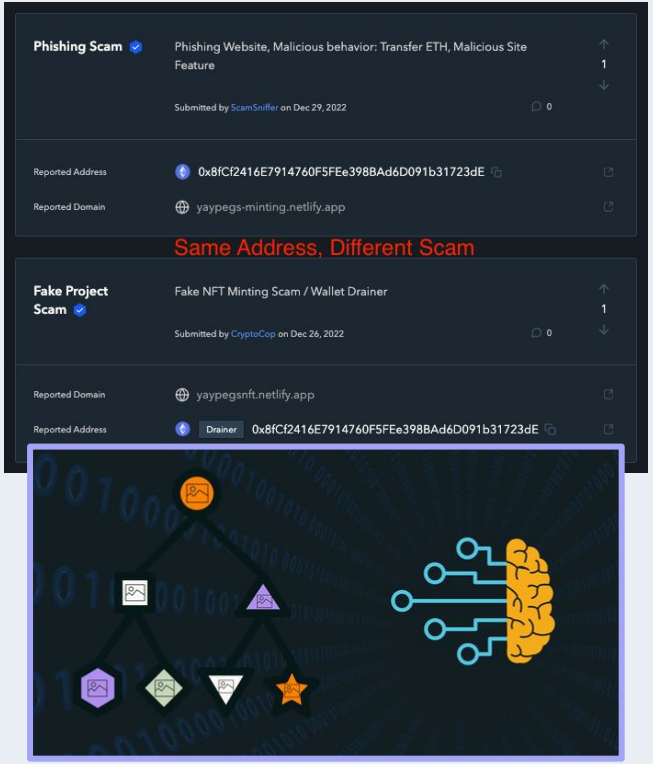
	precision	recall	f1-score	support
ILlicit	0.96	0.98	0.97	599
LICIT	0.94	0.88	0.91	200
accuracy			0.95	799
macro avg	0.95	0.93	0.94	799
weighted avg	0.95	0.95	0.95	799

Multiclass Model Validation Accuracy: 0.7312186978297162
Multiclass Model Classification Report:

	precision	recall	f1-score	support
IMPERSONATION	0.35	0.15	0.21	55
OTHER_SCAM	0.76	0.88	0.81	144
PHISHING	0.64	0.81	0.72	200
Unknown	0.90	0.71	0.79	200
accuracy			0.73	599
macro avg	0.66	0.64	0.63	599
weighted avg	0.73	0.73	0.72	599

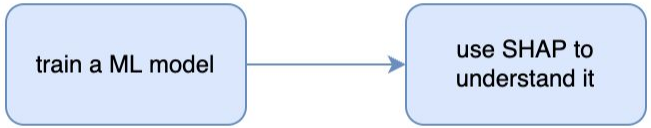
Limitations:
Multi-Categorical Classification

- **Data Imbalance:** Likelihood of predictions. SMOTE faced limitations in addressing the severe 5:1 imbalance in the "IMPERSONATION" model compared to other categories, which had 1000 addresses each for (hence were balanced)
- **Model Overfitting:** Risk of overfitting, which may have resulted in inflated scores. Effective hyperparameter tuning successfully mitigated this issue in initial models.
- **Scalability Hurdles:** Limited categorized ETH addresses online hindered comprehensive model fine-tuning. Emphasizes the need for a larger and more diverse dataset for enhanced training.
- **Data Dependency:** The accuracy of predictions heavily relies on the quality of external input data, especially specific categorization submissions by victims of the frauds.
- **Address Ambiguity:** Certain addresses were found in multiple scam categories. Notably, high model performance suggests specialization in specific scam types.
- **Computation:** Grid search across larger datasets demanded substantial processing time, impacting efficiency.



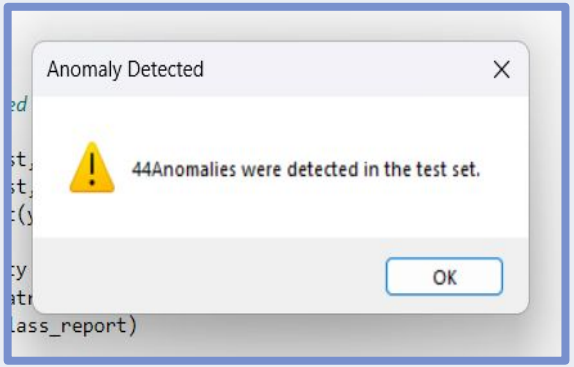
Considerations + Future Work: Multi-Categorical Classification

- **Considerations:**
  - Scraped websites while respecting their terms of service and privacy
- **Future Work:**
  - **Scam Analysis:** Identify key features using SHAP (game theory).
    - ***Adaptability:*** Develop mechanisms for continuous model updates and real-time adaptability (cached models via joblib).
  - **NLP Utilization:** Extract scam type descriptions and messages for enhanced data.
    - ***Regulatory Collaboration:*** Partner with regulators for scam categorization, leveraging NLP models.
  - **Performance Optimization:** Continuously fine-tune model performance using automation and Monte Carlo Simulation for reliability.

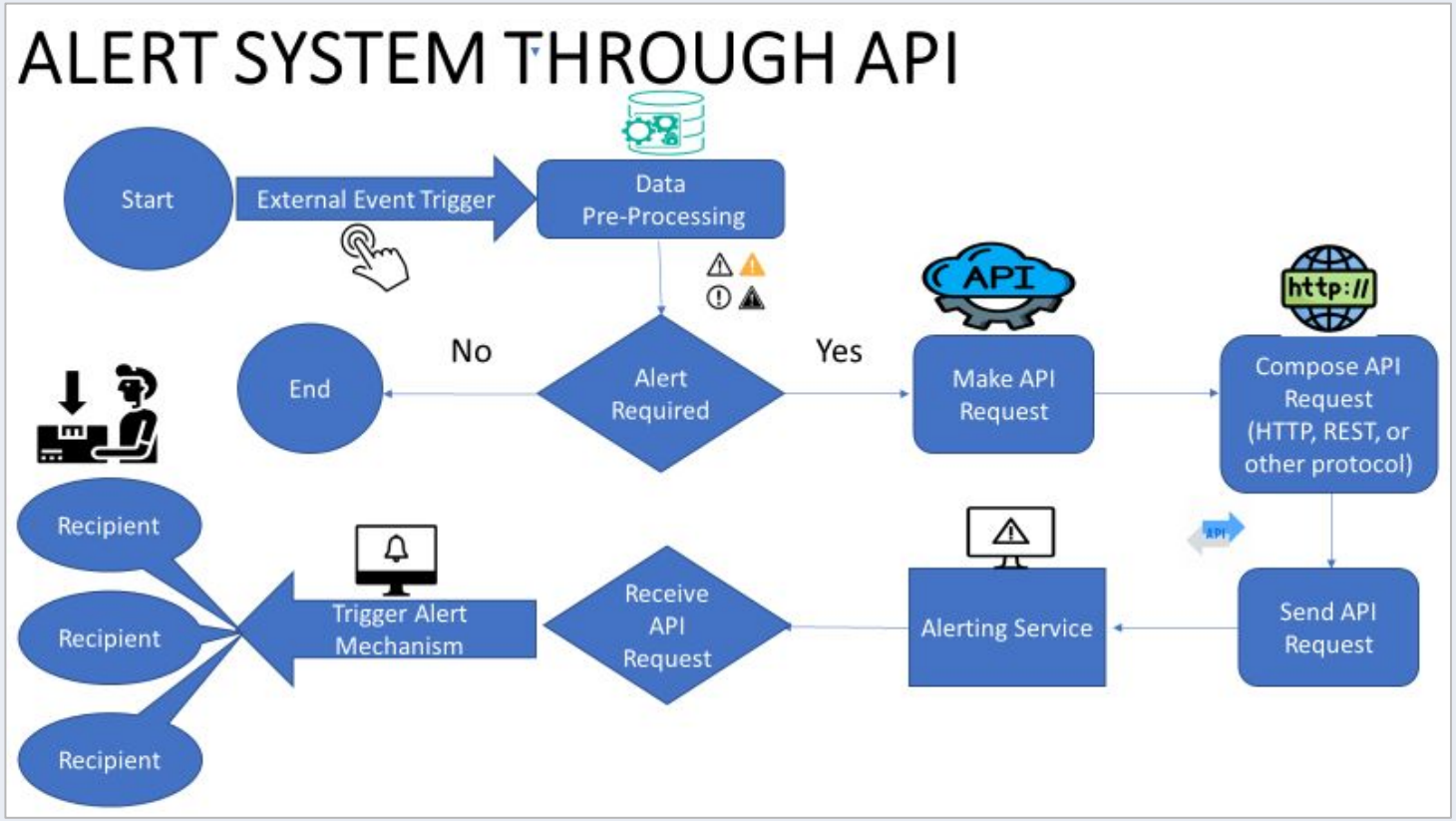


Alert system for anomaly detection

- An alert service for Ethereum fraud detection is crucial for real-time monitoring and early fraud detection in cryptocurrency transactions.
- It enhances security, ensures compliance, and provides customizable alerts, helping users and organizations protect their assets and prevent financial losses.



Flowchart for Alert system



Tech Stack

Geth	Utilized Geth execution client hosted by Quicknode for mempool data	
Python	**Libraries** web3 psycopg2 pandas joblib etherscan-python sklearn Datetime selenium	
PostgreSQL	Google Cloud SQL Database	
Power BI	Statistical Analysis and Visualization Software	

Future Work

- **Explore alternative approaches to classification model creation**
  - Engineer a different array of address features which provide greater coverage of unique token standards and transaction types
  - Train effective ML models that can identify illicit activity on a transactional level
    - Features could be extracted from the transaction data itself, would greatly benefit speed
  - Train multiple ML models for specific types of transactions
    - Simple Ether
    - ERC-20
    - ERC-721
- **Explore alternative real-time model implementation approaches**
  - Experiment with methods of implementation that remove the necessity to make questionable assumptions
- **MEV-Boost Relay Incorporation**
  - Once a working model is trained, incorporate illicit transaction mitigation into MEV-Boost relays or similar system
  - Usher in illicit transaction mitigation to a decentralized blockchain
  - Research ways to mitigate the centralizing effect of using a single database or model for classification

mev-boost

Thank You!

Appendix