



UNIVERSITÀ
DEGLI STUDI
FIRENZE



SCUOLA
ALTI STUDI
LUCCA

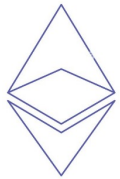
2nd Level Master in
Data Science and Statistical Learning

Exploiting Machine Learning and Blockchain data to Detect Ponzi Schemes on Ethereum

Supervisor
Prof. Fabio Pinelli

Candidate
Luca Pennella

Main goals



The **identification of Ponzi schemes** (type of scam) on the Ethereum blockchain through Machine Learning.



The creation of **publicly available dataset enriched** with data related to the smart contract code.



The use of Explainable Machine Learning for the **identification of the main features** for the detection of Ponzi contracts.

The blockchain



A **Blockchain** is a **shared, immutable ledger** that facilitates the process of recording transactions and tracking assets in a network.



A blockchain **consists of immutable records** organized in blocks, no participant (nodes) can modify or tamper with a transaction after it is recorded.



Transactions are blocked together in an irreversible chain: a blockchain.



Each block carries information from the previous block and transactions are public throughout the network, creating a **secure and trustworthy structure**.

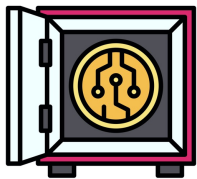
Ethereum: next-generation blockchain



Ethereum is an **open-source public blockchain platform with a computer embedded in it.**



A Smart Contract consist of **software built on the Ethereum** with the goal of automating and decentralizing any kind of application.



Consensus with **Proof-of-stake** leading to better **energy efficiency**, lower **hardware requirements** and greater **decentralization.**

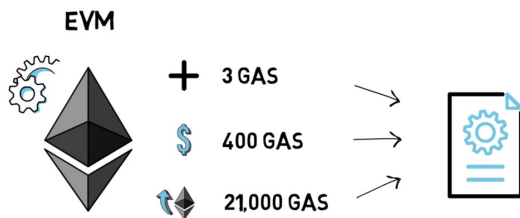
Smart contract journey



The smart contracts are written in the Solidity language.

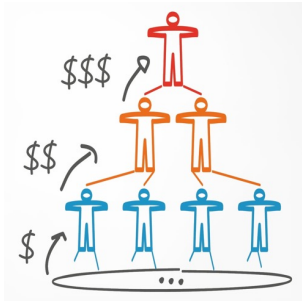


To be executed, the contract must be translated into machine language, in our case into **bytecode**.



Bytecode is a **sequence of commands**, called an **Opcode**. These instructions describe the **step-by-step operations performed by the EVM** (the computer embedded in Ethereum).

From Ponzi Schemes to Smart Ponzi



Ponzi schemes are classic frauds that promise high-yield investment return. The starter of the scheme generates returns for existing investors through revenue paid by new investors.



Smart contracts creates new opportunities for scammers, including the creation of Ponzi schemes, thanks to several attractive features:

- Creators can **stay anonymous**
- smart contracts are **unmodifiable and unstoppable**
- Investors may gain a **false sense of trustworthiness**

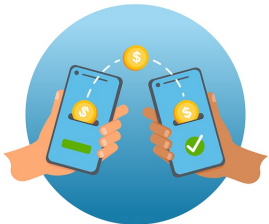
Dataset construction



The dataset is made by **4422 smart contracts**, where 3749 (**85%**) are labelled as **not-Ponzi** and 673 (**15%**) as **Ponzi**.



The main contribution at the dataset level is the **enrichment with data of the code (opcode) linked to the smart contract** that allow the improvement of our classifier.

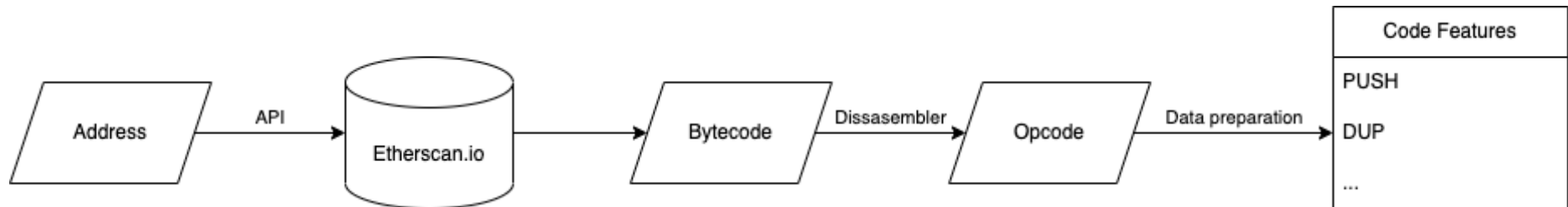


Dataset is made by two types of features:

- **Code Features**
- **Account Features**

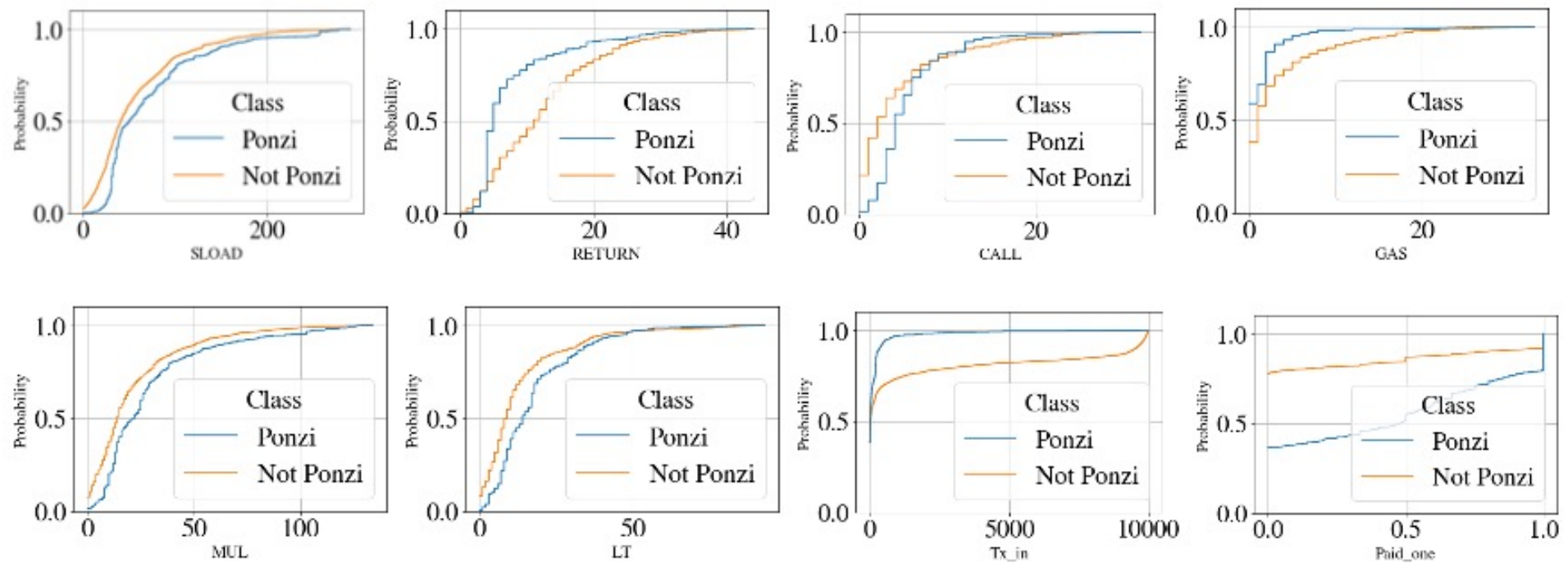
Code Features

The **frequency (absolute and relative) of each type of opcode** for each smart contract is calculated and considered as a features.



With this procedure we **obtained 77 different features**, each feature corresponds to the frequency of a different Opcode within the different contracts analyzed.

Qualitative analysis of the features




The two populations have a **different distribution for many opcodes**, among the **account variables** the ones with differences seem to be Tx_in and Paid_one.

We can hypothesize that these characteristics provide the classifier with a relevant contribution to discriminate between the two classes.

The dataset structure of the experiments performed

We use **7 different dataset** for the project:

1. ***paper***, features used in the work of Chain et al.¹ that use **only 14 account variables**.
-  2. ***full_dataset***, contains the features used by Pinelli and Galletta² which have **developed additional variables** related to transactions (28 features).
3. ***new_dateset***, enriched with all the **variables related to the code** (28 account features + 77 code features).
4. ***only_opcodes***, a dataset with only opcodes.

¹Chen, Weili, et al. "Exploiting blockchain data to detect smart ponzi schemes on Ethereum" *IEEE Acces* 7 (2019)

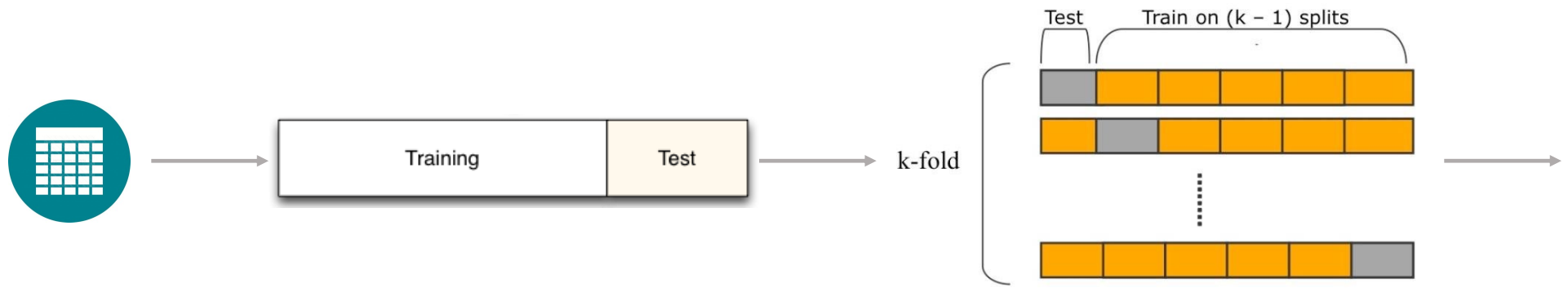
²Galletta, Letterio, and Fabio Pinelli. "Sharpening Ponzi Schemes Detection on Ethereum with Machine Learning" *arXiv preprint* (2023)

The dataset structure of the experiments performed

5. ***new_dataset_best_features***, the best set of variables on *new_dataset* defined by feature selection process (40 variables between account and code features).
6. ***weighted_opcode***, which considers the proportion of each opcode within the contract and not just the frequency plus account features.
7. ***weighted_opcode_best_features***, the best set of variables on *weighted_opcode* defined by feature selection process (35 variables between account and code features).



Data Analytics Pipeline

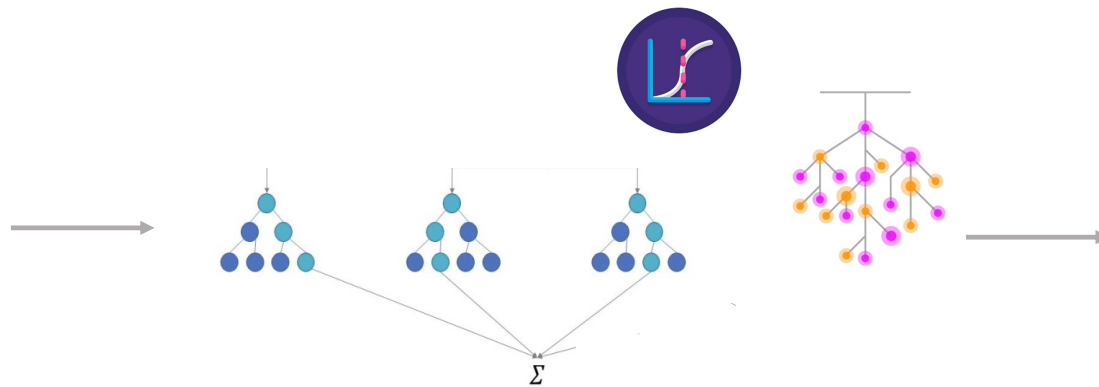


Dataset
Construction.

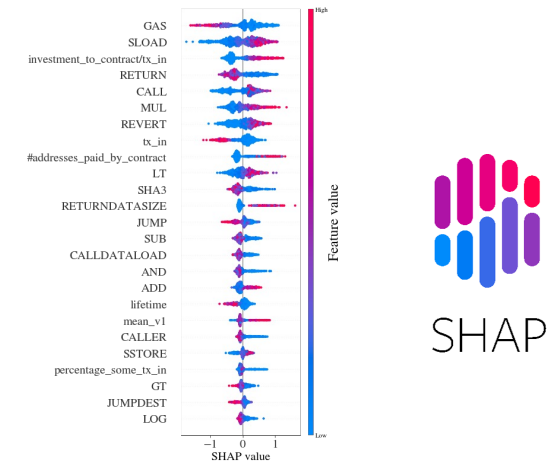
We split the data into an **80% training set** and a **20% test set** stratified on the target variable.

The **cross-validation** splits the **training data into 5 folds**. Then, the model is trained and tested 5 times, varying the fold used as a validation set.

Data Analytics Pipeline



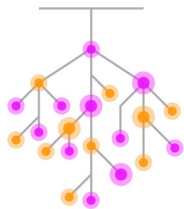
Once we have selected the best values for the hyper-parameters **for each data set and classifier**, we compute the standard evaluation metric.



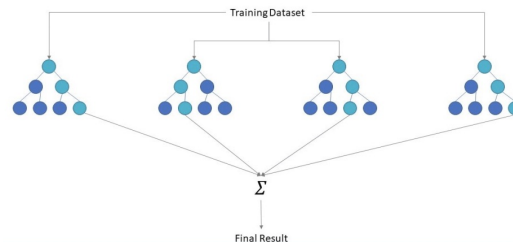
We adopt the eXplainable Artificial Intelligence library SHAP to visually investigate how the most important features impact the classification process.

Algorithms used

We used Tree-Based Models to solve the classification problem:



Decision Tree



Random Forest



LightGBM

We used the grid search technique to get the best hyper-parameters:

```
'classifier': [DecisionTreeClassifier(random_state=42)],  
'classifier__criterion' : ['entropy', 'gini'],  
'classifier__max_depth' : [5, 6, 7, 8, 9],  
'classifier__min_samples_split' : [5, 10, 15],  
'classifier__max_features' : range(6,X_train.shape[1],3),  
'classifier__class_weight' : [{0:1, 1:4}, {0:1, 1:4.5}]
```

```
'classifier': [RandomForestClassifier(random_state=42)],  
'classifier__n_estimators': [150, 175, 200, 225, 250],  
'classifier__min_samples_split' : [5, 15, 30],  
'classifier__criterion': ['gini', 'entropy'],  
'classifier__class_weight' : [{0:1, 1:5.5}],  
'classifier__bootstrap' : [True, False]
```

```
'classifier': [lgb.LGBMClassifier(boosting_type='gbdt', n_jobs=4,  
                                importance_type='split', random_state=42)],  
'classifier__learning_rate' : [0.1, 0.01],  
'classifier__n_estimators' : [80, 100, 120],  
'classifier__max_depth' : [10, 15, 20],  
'classifier__colsample_bytree' : [0.5, 0.8, 1],  
'classifier__reg_alpha' : [0, 0.1, 0.2],  
'classifier__reg_lambda' : [1, 10, 15]]
```

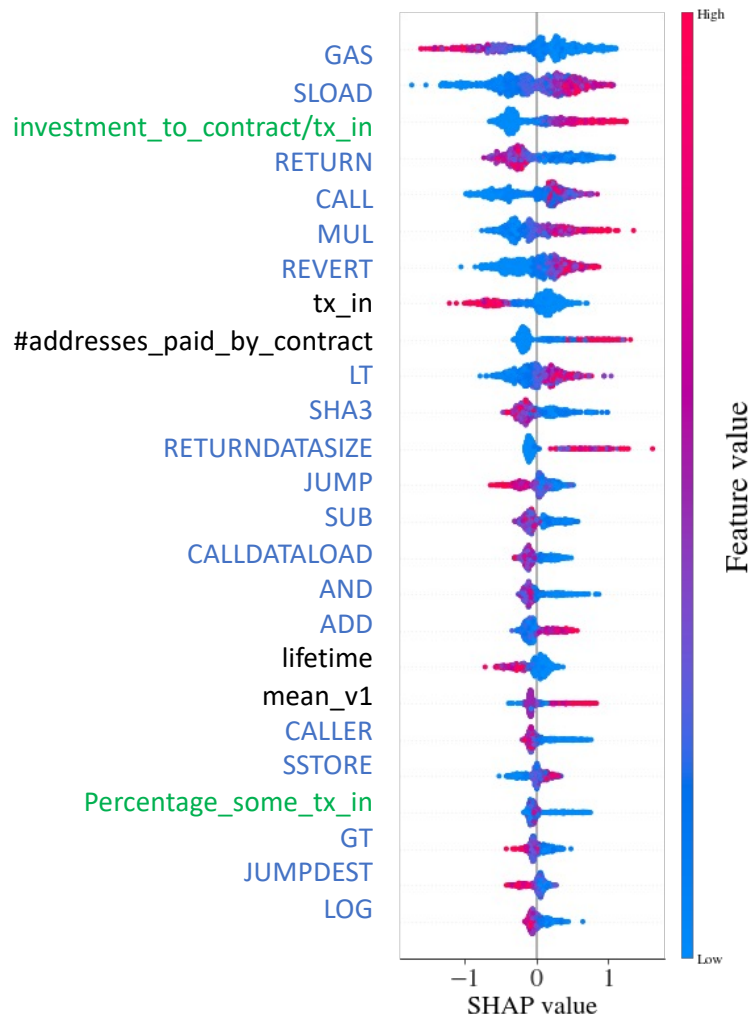
Result

We used Tree-Based Models to solve the classification problem:

Best Model	AUC	PRECISION	RECALL	ACCURACY	F1
Paper	0,794	0,776	0,333	0,884	0,466
full_dataset	0,885	0,842	0,474	0,906	0,607
new_dataset	0,978	0,911	0,756	0,951	0,826
only_opcode	0,979	0,919	0,756	0,953	0,829
new_dataset_best_features	0,979	0,927	0,748	0,953	0,828
weighted_opcode	0,982	0,921	0,777	0,956	0,843
weighted_opcode_best_features	0,976	0,879	0,756	0,947	0,813

We can see an **important improvement in all evaluation metrics**. As for the methods used, the performances are quite similar.

XAI



Through SHAP it is possible to detect the importance of the features, as we can see in the image both the **account variables and those of the code are among the twenty most relevant.**

The variables implemented in our project are highlighted in blue, those of Pinelli and Galletta in green and the remaining ones of the starting paper of Chain et al.

Conclusion

Achieved results

- We have highlighted how the use of additional features combined with complex classification algorithms has **improved performance**.
- The explanation techniques allow us to make the work much **more understandable** even to external listeners.

Further improvements

- **Extension of the dataset** with additional types of scams.
- Creation of an **ensemble of models** with the aim of further **improving the performances**.
- **Improved problem explainability** using additional Explainable Artificial Intelligence techniques. Among those identified: dlime, Anchors, PDP and Ice.

Thank you very much for your attention!