EtherScan

Obiettivo del Progetto

Il presente progetto mira a rilevare possibili operazioni fraudolente sulla blockchain di Ethereum, partendo dall'analisi dei movimenti di un singolo account. L'obiettivo è individuare schemi anomali o comportamenti sospetti che possano indicare attività illecite.

L'esigenza di identificare transazioni sospette nasce dalla crescente diffusione delle criptovalute e dall'aumento di fenomeni come il riciclaggio di denaro e le truffe finanziarie. Per affrontare questo problema, è stato sviluppato un modello di Machine Learning in grado di analizzare i dati delle transazioni e distinguere tra operazioni legittime e potenzialmente fraudolente.

Nelle sezioni successive vengono illustrate le varie fasi di sviluppo del progetto: dalla raccolta e preprocessamento dei dati fino alla valutazione del modello, con una particolare attenzione alla precision e all'affidabilità delle previsioni.

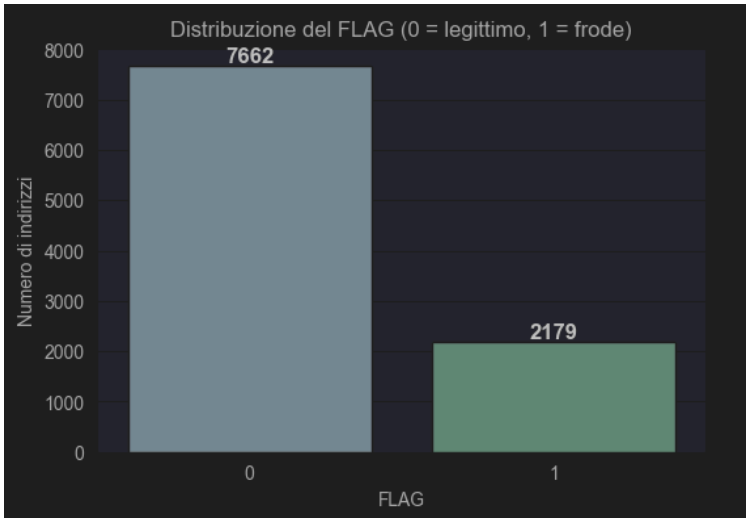
Analisi del Dataset

Il dataset utilizzato per questo studio è stato reperito su Kaggle al seguente link: [Ethereum Fraud Detection Dataset]().

- Composizione: È composto da 50 parametri per ogni riga e include già le etichette che indicano le transazioni fraudolente.
- Finalità: Costituisce un'ottima base per l'addestramento di un modello di classificazione e fornisce una ricca varietà di informazioni.

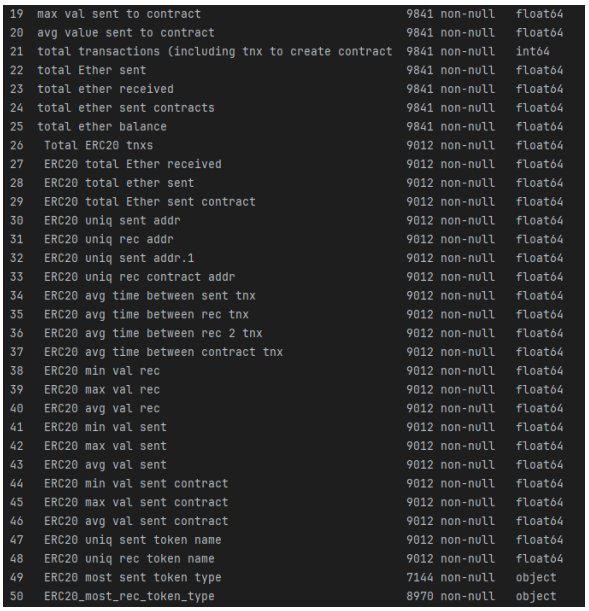
Sbilanciamento dei Dati

Il dataset risulta sbilanciato, con 7662 transazioni classificate come non fraudolente (Flag = False) e 2179 transazioni classificate come fraudolente (Flag = True). Per gestire questo problema, è stato adottato un approccio a livello algoritmico basato sulla modifica dei pesi delle classi, aumentando quello della classe meno rappresentata. In questo modo, il modello tende ad apprendere in modo più equilibrato, senza trascurare i casi di frode.



Gestione dei Valori Nulli

Un aspetto critico riguarda la presenza di valori nulli, localizzati esclusivamente nelle transazioni prive di operazioni ERC20. Dopo un'analisi approfondita, si è stabilito che tali valori mancanti non corrispondono a veri e propri errori, ma riflettono l'assenza di interazioni ERC20. Pertanto, sono stati sostituiti con 0, interpretando correttamente la mancanza di movimenti di questo tipo.



```
19  max val sent to contract                          9841 non-null   float64
20  avg value sent to contract                        9841 non-null   float64
21  total transactions (including tnx to create contract  9841 non-null   int64
22  total Ether sent                                  9841 non-null   float64
23  total ether received                              9841 non-null   float64
24  total ether sent contracts                        9841 non-null   float64
25  total ether balance                               9841 non-null   float64
26   Total ERC20 tnxs                                 9012 non-null   float64
27   ERC20 total Ether received                       9012 non-null   float64
28   ERC20 total ether sent                           9012 non-null   float64
29   ERC20 total Ether sent contract                  9012 non-null   float64
30   ERC20 uniq sent addr                             9012 non-null   float64
31   ERC20 uniq rec addr                              9012 non-null   float64
32   ERC20 uniq sent addr.1                           9012 non-null   float64
33   ERC20 uniq rec contract addr                     9012 non-null   float64
34   ERC20 avg time between sent tnx                  9012 non-null   float64
35   ERC20 avg time between rec tnx                   9012 non-null   float64
36   ERC20 avg time between rec 2 tnx                 9012 non-null   float64
37   ERC20 avg time between contract tnx              9012 non-null   float64
38   ERC20 min val rec                                9012 non-null   float64
39   ERC20 max val rec                                9012 non-null   float64
40   ERC20 avg val rec                                9012 non-null   float64
41   ERC20 min val sent                               9012 non-null   float64
42   ERC20 max val sent                               9012 non-null   float64
43   ERC20 avg val sent                               9012 non-null   float64
44   ERC20 min val sent contract                      9012 non-null   float64
45   ERC20 max val sent contract                      9012 non-null   float64
46   ERC20 avg val sent contract                      9012 non-null   float64
47   ERC20 uniq sent token name                       9012 non-null   float64
48   ERC20 uniq rec token name                        9012 non-null   float64
49   ERC20 most sent token type                       7144 non-null   object
50   ERC20_most_rec_token_type                        8970 non-null   object
```

Outlier e Normalizzazione

Durante l'analisi esplorativa, sono stati individuati numerosi outlier: valori che si discostano in modo significativo dal resto dei dati, talvolta presenti una sola volta. Questi outlier possono causare overfitting, poiché il modello rischia di adattarsi a caratteristiche troppo specifiche.

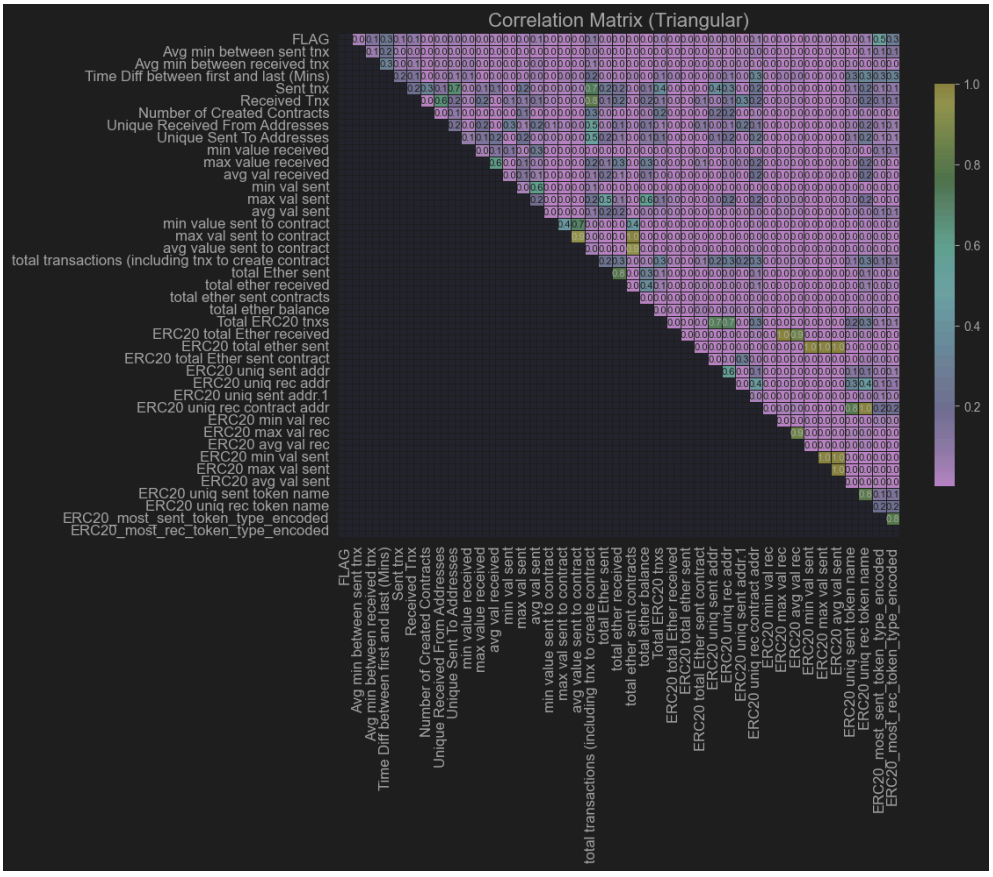
Per attenuare tale rischio e migliorare la convergenza degli algoritmi, è stata effettuata una scalatura e normalizzazione dei dati tramite la funzione StandardScaler di `sklearn`. Questa procedura riduce l'effetto dei valori estremi e semplifica il processo di apprendimento.

Feature Engineering

In questa fase, ci si concentra sull'identificazione, la rimozione o la creazione di feature (variabili) che possano influenzare positivamente la capacità del modello di rilevare attività fraudolente.

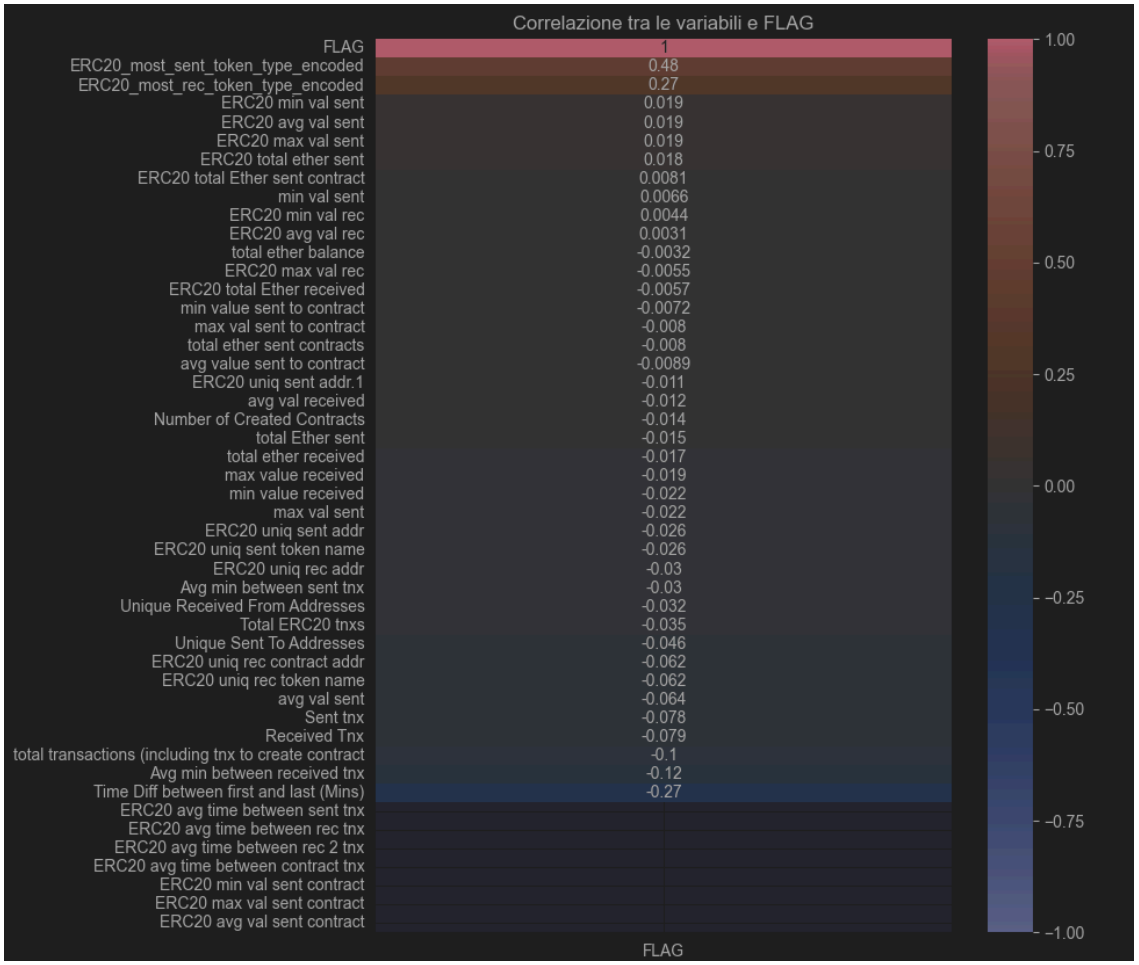
Selezione delle Feature

- **Ridondanza**: Alcune feature con correlazione superiore al 90% sono state rimosse, poiché introducevano eccessiva ridondanza e aumentavano il rumore, senza fornire informazioni nuove.



Rimozione di Colonne per Evitare Overfitting (Data Leakage)

- **Colonne eliminate:** `ERC20_most_sent_token_type_encoded` e `ERC20_most_rec_token_type_encoded`.
- **Motivazione:** Durante una fase di test iniziale, l'inclusione di tali colonne ha portato a risultati insolitamente elevati (F1 score prossimo al 99%), rivelandosi un chiaro caso di overfitting. Il modello apprendeva la relazione diretta tra un token specifico e la frode, compromettendo la capacità di generalizzare a nuovi token. Pertanto, si è preferito rimuoverle.

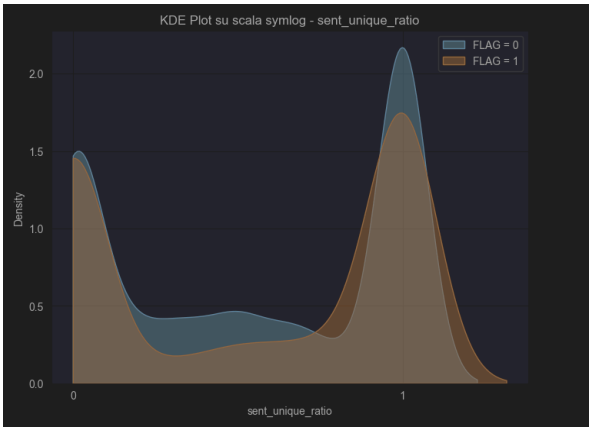


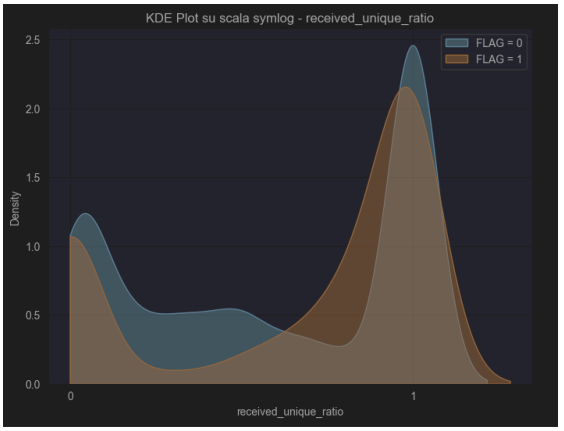
Creazione di Nuove Feature

Per arricchire il dataset con variabili più informative, sono state introdotte nuove feature:

- **Rapporto tra indirizzi unici e transazioni inviate:** Indica la diversificazione degli indirizzi di destinazione rispetto al numero di transazioni totali inviate.
- **Rapporto tra indirizzi unici e transazioni ricevute:** Rappresenta la varietà degli indirizzi di provenienza, misurando il livello di dispersione delle fonti.

L'obiettivo di queste trasformazioni è ridurre la probabilità di overfitting e migliorare la robustezza del modello, fornendo informazioni più specifiche sul comportamento delle transazioni.





Tramite un'analisi dei grafici, possiamo notare come in realtà queste feature non danno i risultati sperati in quanto non aggiungono informazioni abbastanza rilevanti da permettere al modello di migliorare, per questo motivo sono state successivamente rimosse.

Sviluppo del Modello

La scelta è ricaduta su un Random Forest, un modello di ensemble capace di combinare i risultati di molti alberi decisionali. Questa strategia riduce il rischio di overfitting legato all'utilizzo di un singolo albero e migliora sensibilmente le prestazioni.

Motivi della Scelta

- **Semplicità di Utilizzo:** Rispetto ad altre tecniche, le Random Forest risultano intuitive e veloci da addestrare.
- **Buone Prestazioni:** Le performance ottenute con un singolo albero decisionale erano già discrete, ma l'adozione di un ensemble ha incrementato ulteriormente la precisione.

Risultati Iniziali

Nella configurazione iniziale, la Random Forest ha raggiunto circa il 97% di precision e l'83% di recall, riuscendo a discriminare efficacemente le transazioni fraudolente da quelle legittime.

Ottimizzazione della Soglia

Per massimizzare la recall, è stata abbassata la soglia di probabilità oltre la quale una transazione viene considerata fraudolenta. In particolare, impostando la soglia intorno al 30%, si è ottenuto un compromesso che ha portato a:

- **Precision: 91%**
- **Recall: 89%**

Riducendo la soglia, il modello aumenta la capacità di riconoscere transazioni sospette (riducendo i falsi negativi), a costo di una maggiore probabilità di generare falsi positivi. Tuttavia, in un settore come quello delle frodi, è preferibile segnalare qualche falso allarme piuttosto che rischiare di non intercettare attività illecite.

Conclusioni

Le analisi e gli esperimenti condotti dimostrano l'efficacia dell'approccio scelto nell'individuazione di operazioni fraudolente sulla blockchain di Ethereum. La combinazione di:

1. Accurate Operazioni di Preprocessing (gestione valori nulli, normalizzazione e rimozione outlier)
2. Feature Engineering Mirata (riduzione della ridondanza)
3. Modello Random Forest con tecniche di ensemble e ottimizzazione della soglia

ha consentito di ottenere un buon equilibrio tra precision e recall, dimostrandosi solida anche in contesti dove la mancanza di dati bilanciati e l'elevata presenza di outlier rappresentano criticità significative. Nell’eventualità del deploy del modello, si potrebbero fare ulteriori analisi con stakeholder per valutare se migliorare la precision piuttosto che la recall.