tinexta
defence

NetFlowMeter: analisi del dataset CICIDS2017 e rilevamento delle intrusioni tramite ML

#TinextaDefenceBusiness

AI4Cyber

AI4Cyber studio, è il nuovo spazio di ricerca applicata di Tinexta Defence, dedicato alla ricerca e applicazione dell'**Intelligenza Artificiale in ambito cybersecurity**.

Al suo interno, diversi specialisti con competenze ed esperienze eterogenee collaborano per affrontare le sfide poste dal panorama delle minacce informatiche, accompagnare i clienti nell'adozione dell'AI nei propri processi aziendali e condividere le attività di ricerca con la comunità tecnico-scientifica.

Il team cura l'**intero ciclo di sviluppo dei sistemi basati su AI**: dalla raccolta e dal preprocessing dei dati, all'addestramento e validazione dei modelli, fino al deployment in produzione.

Le soluzioni proposte si fondano su tecnologie avanzate di **Machine Learning**, **Deep Learning** e **Large Language Models**, e possono essere personalizzate in base alle specifiche esigenze del cliente.

L'AI Team è costantemente impegnato in attività di ricerca e sperimentazione, con un'attenzione particolare agli **aspetti etici**, alla **trasparenza** e alla **tutela della privacy**.

L'obiettivo è proporre soluzioni innovative che siano in linea con i principi di **equità, inclusività e rispetto dei diritti fondamentali**.

Summary

Abstract

Il presente studio analizza due varianti del dataset di traffico di rete CICIDS2017. La prima variante si basa sui file CSV di una versione revisionata del dataset originale; la seconda, invece, richiede la generazione dei flussi di rete a partire dai file PCAP grezzi, utilizzando NetFlowMeter, uno strumento sviluppato internamente per superare alcune limitazioni di CICFlowMeter, generatore di flussi ampiamente adottato in ambito accademico. Lo studio si articola in due fasi principali: analisi esplorativa e rilevamento di anomalie.

L'analisi esplorativa, condotta tramite alberi decisionali, ha evidenziato risultati anomali anche con una profondità massima dell'albero notevolmente ridotta. Questo comportamento suggerisce la presenza di feature fortemente discriminanti nel dataset, in particolare legate ai flag TCP RST e PSH e alla lunghezza dei pacchetti provenienti dal server. L'analisi con Wireshark ha inoltre rivelato criticità aggiuntive, tra cui l'interpretazione errata dei pacchetti ARP da parte di CICFlowMeter.

Nella seconda fase, è stato implementato un sistema di rilevamento delle anomalie semi-supervisionato, basato su un autoencoder addestrato esclusivamente su traffico normale. I risultati ottenuti mostrano un elevato tasso di falsi positivi, che conduce inevitabilmente a un sovraccarico di allarmi (alert fatigue). Il lavoro propone quindi alcune strategie per mitigare questo effetto.

Autori:

- Antonio Repola: Data Scientist
- Simona Sorgente: Team Leader AI4Cyber

1. Introduzione

I sistemi di rilevamento e prevenzione delle intrusioni (IDS/IPS) sono strumenti di sicurezza informatica fondamentali per la protezione delle reti da accessi non autorizzati e traffico malevolo. L'evoluzione delle minacce informatiche ha ridotto l'efficacia dei metodi tradizionali basati su firme, favorendo quindi l'adozione di tecniche di machine learning (ML). Queste tecniche operano generalmente su aggregazioni di pacchetti in flussi di rete, ottenibili a partire da file PCAP mediante strumenti dedicati. Un flusso di rete è definito come una sequenza di pacchetti scambiati tra una sorgente e una destinazione entro una finestra temporale specifica. È caratterizzato da cinque attributi principali: indirizzi IP e porte sorgente e destinazione, nonché il protocollo di rete utilizzato.

NetFlowMeter è uno strumento per la generazione di flussi di rete bidirezionali, in grado di elaborare file PCAP ed estrarre un set di feature utilizzabili in applicazioni di ML. Si tratta di una reingegnerizzazione di CICFlowMeter [1], uno strumento ampiamente adottato in ambito accademico per lo stesso scopo, con l'obiettivo di migliorarne le prestazioni e correggere alcuni bug che compromettono la qualità dei dataset prodotti. La versione 1.0 di NetFlowMeter, compatibile con CICFlowMeter, è disponibile su GitHub con licenza MIT [2]. Tuttavia, nello studio in questione è stata utilizzata la versione 2.0, attualmente in fase di sviluppo. Questa versione rinuncia alla compatibilità con CICFlowMeter per consentire la correzione di ulteriori bug e l'aggiunta di nuove feature. Maggiori dettagli sullo strumento sono disponibili nel documento pubblicato in precedenza [3].

2. Dataset di riferimento

La seguente analisi illustra il processo di esplorazione del dataset CICIDS2017 [4, 5] finalizzato all'identificazione di diverse problematiche in esso contenute. La scelta di questo dataset è motivata da diverse ragioni. In primo luogo, si tratta di una raccolta pubblica di dati relativi a traffico di rete, specificamente progettata per la valutazione di IDS basati su ML. In secondo luogo, questo dataset è ampiamente riconosciuto e adottato come benchmark nella letteratura scientifica di settore. Inoltre, sono disponibili sia le catture del traffico (in formato PCAP) sia i flussi di rete (in formato CSV) generati utilizzando CICFlowMeter; i file PCAP sono necessari per la generazione di flussi tramite NetFlowMeter. Infine, il dataset rappresenta un caso di studio significativo per l'analisi della qualità dei dati, in quanto presenta alcune criticità già note alla comunità scientifica.

L'acquisizione dei dati contenuti in CICIDS2017 è avvenuta nell'arco di cinque giorni nel luglio 2017. La giornata di lunedì è caratterizzata esclusivamente da traffico benigno, mentre dal martedì al venerdì si osserva una combinazione di traffico benigno e scenari di attacco specifici.

Giorno	**Attività del mattino**	**Attività del pomeriggio**
Lunedì	Solo traffico benigno	
Martedì	FTP brute force	SSH brute force
Mercoledì	DoS	Heartbleed
Giovedì	Attacchi Web	Infiltration
Venerdì	Botnet ARES	Port scan, DDoS LOIC

In seguito, il dataset è stato analizzato da un altro gruppo di ricercatori [6]. Tale analisi ha rivelato numerosi errori precedentemente non documentati, presenti in tutte le fasi del ciclo di creazione del dataset: orchestrazione degli attacchi, generazione delle feature, documentazione ed etichettatura. Questi errori sollevano dubbi sulle conclusioni di numerosi studi che hanno utilizzato il dataset come riferimento. In risposta alle criticità emerse, i ricercatori hanno realizzato e reso disponibile una versione aggiornata del dataset, la cui ultima modifica risale al 27 aprile 2023 [7].

Sia l'etichettatura originale che quella revisionata attestano che il dataset è fortemente sbilanciato, il che rappresenta un ostacolo al suo utilizzo nell'addestramento di modelli di ML. Di seguito sono elencate alcune delle etichette del dataset aggiornato e il loro numero di occorrenze.

BENIGN	1582566
Portscan	159066
DoS Hulk	158468
DDoS	95144
Infiltration – Portscan	71767
DoS GoldenEye	7567
Botnet – Attempted	4067
Infiltration	36
Web Attack – XSS	18
Web Attack – SQL Injection	13
Heartbleed	11

Per condurre l'analisi, si è deciso di utilizzare due varianti del dataset CICIDS2017:

1. La variante CICFlowMeter, basata sui file CSV della versione revisionata del dataset originale.
2. La variante NetFlowMeter (2.0), che ha richiesto la generazione dei flussi in formato CSV a partire dai file PCAP originali utilizzando il nuovo strumento.

Successivamente, si è proceduto alla rietichettatura dei dati per suddividerli in due sole classi: BENIGN per i flussi benigni e ATTACK per quelli malevoli.

Per la variante CICFlowMeter, è sufficiente collassare le diverse classi di attacco (Port scan, DoS Hulk, DDoS, ecc.) in un'unica macro-classe (ATTACK). Per la variante NetFlowMeter, invece, l'etichettatura è stata effettuata facendo riferimento alla procedura definita nella documentazione della versione revisionata del dataset [8]. È stata inoltre effettuata una pulizia dei dati, che ha comportato la rimozione di righe contenenti valori infiniti, presenti esclusivamente nella variante CICFlowMeter.

Si fa notare che l'uso di NetFlowMeter comporta una riduzione del numero complessivo di flussi rispetto a CICFlowMeter, dovuta all'eliminazione della logica di troncamento di quest'ultimo, che prevede il troncamento dei flussi ogni 2 minuti.

	CICFlowMeter	**NetFlowMeter**
BENIGN	1594540	1353140
ATTACK	505431	497999

3. Analisi esplorativa

Una volta che i dati sono stati etichettati, è possibile procedere con la fase di analisi esplorativa, per la quale esistono molteplici tecniche e strumenti. Tuttavia, in quest'occasione si vuole focalizzare l'attenzione su come utilizzare gli alberi decisionali per estrarre informazioni sulla qualità dei dati e identificare feature importanti.

Gli alberi decisionali sono una tecnica di apprendimento supervisionato per la classificazione e la regressione, e producono strutture ad albero basate su regole facilmente interpretabili: partendo dalla radice, l'albero valuta una sequenza di condizioni sulle feature fino a raggiungere una foglia che fornisce la previsione.

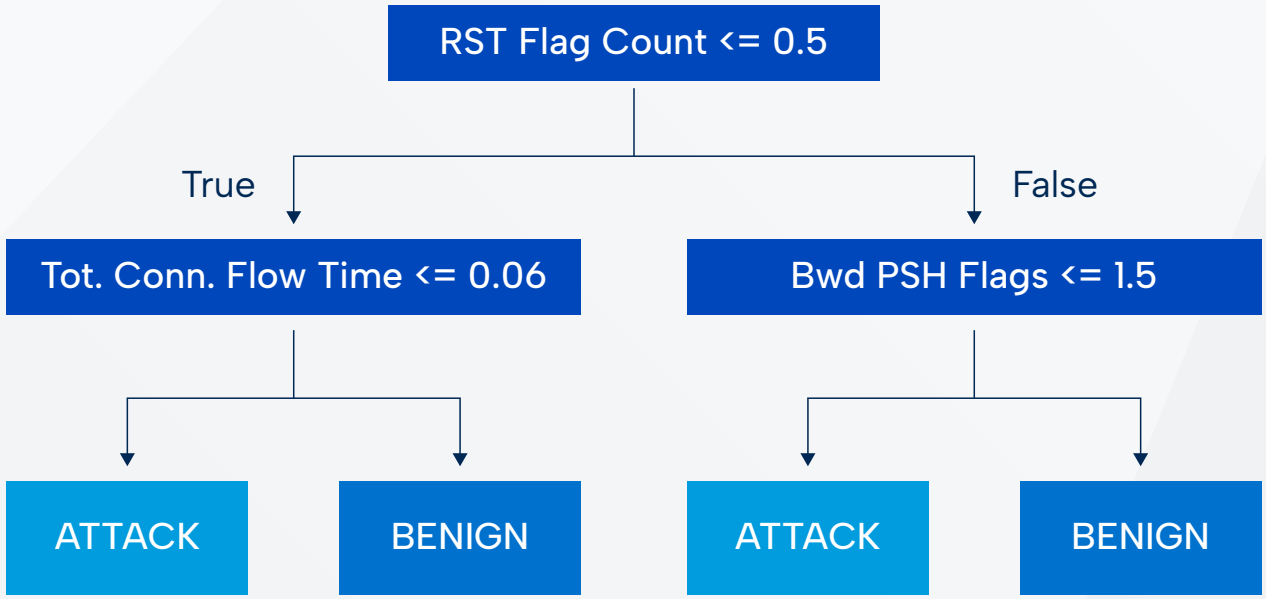

Figura 1: Esempio di albero di decisione

Si è proceduto quindi alla costruzione di un classificatore binario usando la classe `DecisionTreeClassifier` di scikit-learn [9], al fine di distinguere tra flussi benigni e malevoli.

Per l'addestramento, sono state escluse alcune colonne come l'identificatore del flusso, gli indirizzi IP, le porte e il timestamp. Pertanto, il numero di feature è pari a 78 per la variante CICFlowMeter e 86 per quella NetFlowMeter. I dati sono stati quindi suddivisi in set di addestramento (80%) e test (20%). Inizialmente, è stata impostata una profondità massima dell'albero (parametro max_depth) pari a 5.

È possibile visualizzare i risultati a valle dell'addestramento sul set di test utilizzando una matrice di confusione, che mostra le prestazioni del modello confrontando le previsioni con le classi effettive. La diagonale principale indica i veri negativi (*true negative, TN*) e i veri positivi (*true positive, TP*), ovvero le risposte corrette del modello, mentre le altre due celle indicano gli errori. I falsi negativi (*false negative, FN*) rappresentano i flussi malevoli non riconosciuti come tali, mentre i falsi positivi (false positive, FP) corrispondono a falsi allarmi. Di seguito è riportata la matrice di confusione per il set di test della variante CICFlowMeter, che mostra risultati promettenti.

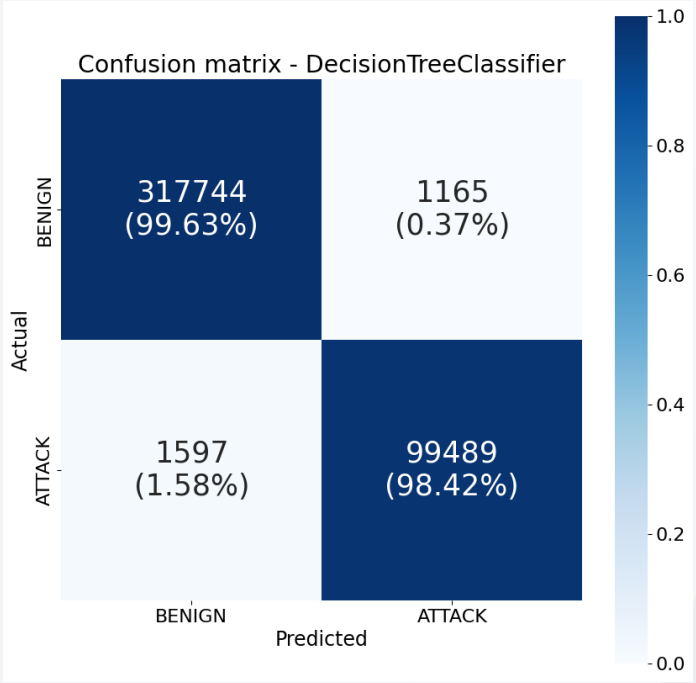

Figura 2: Matrice di confusione per il set di test della variante CICFlowMeter, max_depth=5

A seguire, invece, la matrice di confusione per il set di test della variante NetFlowMeter, che evidenzia un lieve miglioramento delle prestazioni.

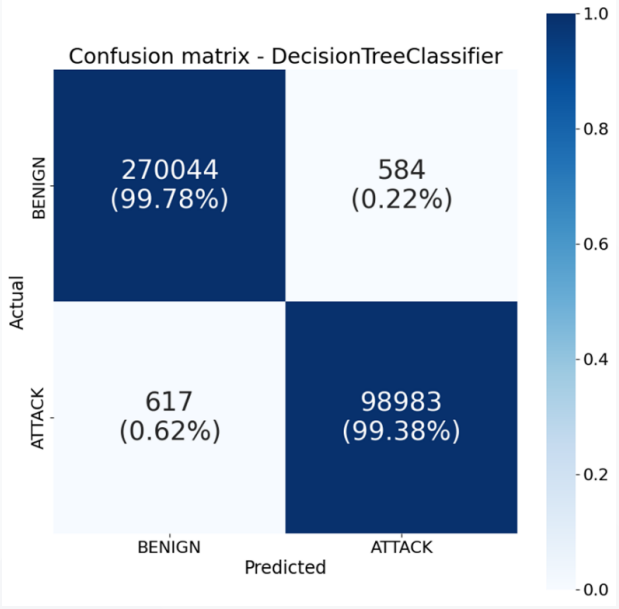

Figura 3: Matrice di confusione per il set di test della variante NetFlowMeter, max_depth=5

Riducendo la profondità massima dell'albero a 3 e ripetendo l'addestramento si ottengono risultati ancora soddisfacenti per entrambi i set di test, il che desta sospetti.

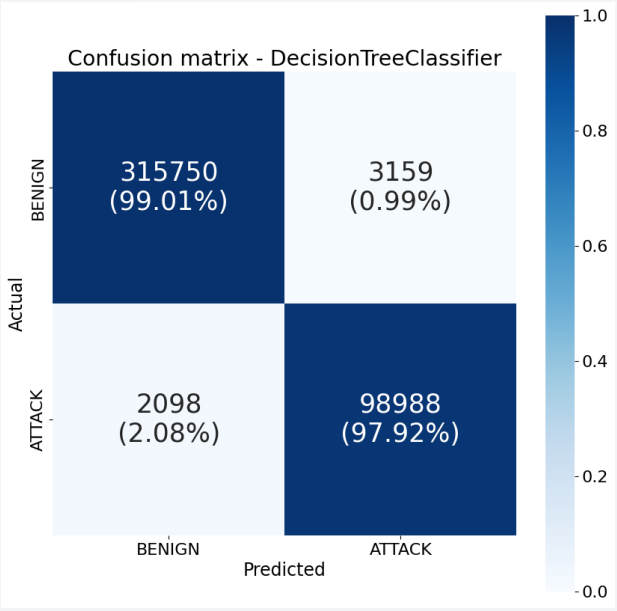

Figura 4: Matrice di confusione per il set di test della variante CICFlowMeter, max_depth=3

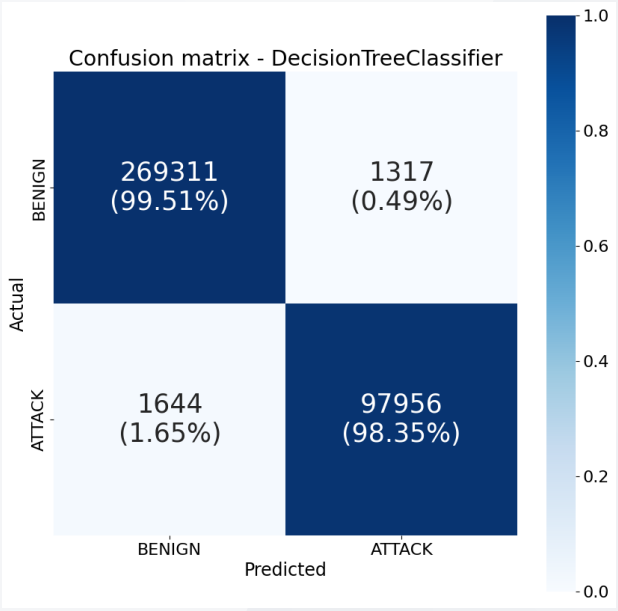

Figura 5: Matrice di confusione per il set di test della variante NetFlowMeter, max_depth=3

È possibile determinare l'importanza delle feature utilizzando, ad esempio, la tecnica permutation importance [10]. È interessante notare che le tre feature più importanti sono le stesse in entrambi i casi. Inoltre, è singolare che la prima e la terza feature siano relative a flag del TCP, in particolare RST (Reset) e PSH (Push).

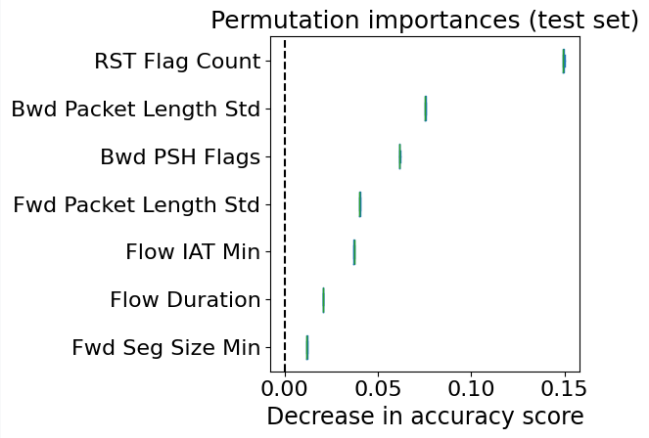

Figura 6: Importanza delle feature, variante CICFlowMeter

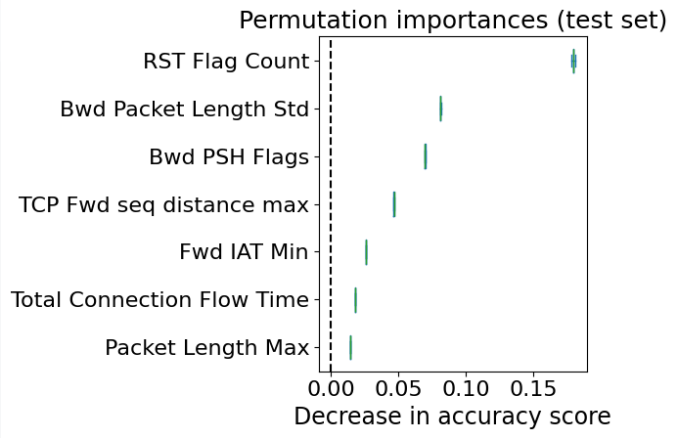

Figura 7: Importanza delle feature, variante NetFlowMeter

D'ora in avanti, per semplicità, verranno mostrati solo i risultati relativi alla variante NetFlowMeter. Con una riduzione ulteriore della profondità massima a 2, si osserva un lieve calo delle prestazioni, che tuttavia restano complessivamente soddisfacenti.

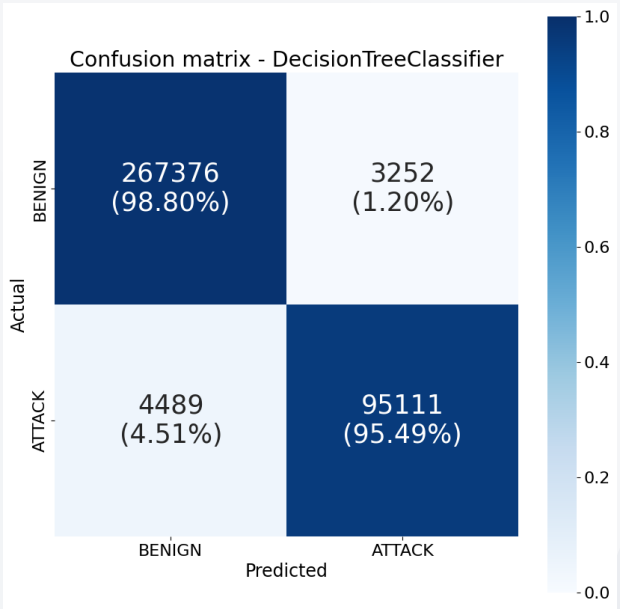

Figura 8: Matrice di confusione per il set di test della variante NetFlowMeter, max_depth=2

Poiché l'albero decisionale è ora di piccole dimensioni, è possibile esaminarne la rappresentazione testuale. In sostanza, se il numero di pacchetti con il flag RST impostato è pari a zero, si osserva la durata della connessione e se questa tende a zero si prevede un attacco. Se, invece, il conteggio dei flag RST è superiore a zero, si osserva il numero di volte in cui il flag PSH è impostato nei pacchetti inviati dal server e se questo è pari a 0 oppure 1 si prevede un attacco. Questo comportamento insolito potrebbe essere attribuibile allo squilibrio tra le classi.

```
|--- RST Flag Count <= 0.50
|   |--- Total Connection Flow Time <= 0.06
|   |   |--- class: ATTACK
|   |--- Total Connection Flow Time >  0.06
|   |   |--- class: BENIGN
|--- RST Flag Count >  0.50
|   |--- Bwd PSH Flags <= 1.50
|   |   |--- class: ATTACK
|   |--- Bwd PSH Flags >  1.50
|   |   |--- class: BENIGN
```

Infine, è possibile impostare la profondità massima dell'albero a 1, ottenendo una semplice regola if-else sulla feature RST Flag Count. Le prestazioni sul set di test risultano ridotte, ma non drasticamente. È importante evidenziare che il flag RST viene utilizzato per terminare bruscamente una sessione TCP. Pertanto, l'interpretazione della regola è che ogni volta che una connessione non viene chiusa correttamente si verifica un attacco. In considerazione delle prestazioni ottenute, tale comportamento risulta alquanto sospetto.

```
|--- RST Flag Count <= 0.50
|   |--- class: BENIGN
|--- RST Flag Count >  0.50
|   |--- class: ATTACK
```

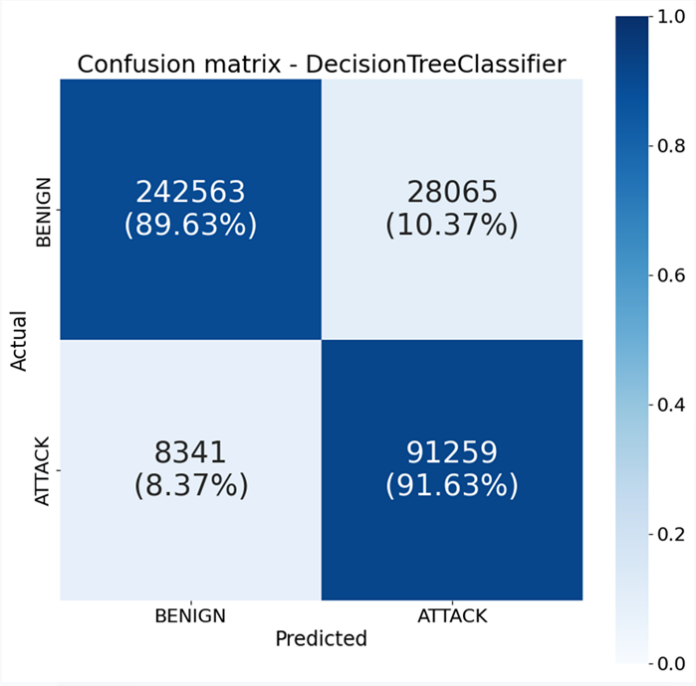

Figura 9: Matrice di confusione per il set di test della variante NetFlowMeter, max_depth=1

Se si considera solo il traffico del lunedì e del mercoledì, ovvero il giorno dedicato agli attacchi Denial-of-Service (DoS), e si ripete l'addestramento, si osserva un ulteriore comportamento degno di nota. In questo caso, si ottiene una regola if-else sulla feature Bwd Packet Len Std che rappresenta la deviazione standard della lunghezza dei pacchetti provenienti dal server. Le prestazioni sul set di test risultano soddisfacenti.

```
|--- Bwd Packet Len Std <= 1496.86
|   |--- class: BENIGN
|--- Bwd Packet Len Std >  1496.86
|   |--- class: ATTACK
```

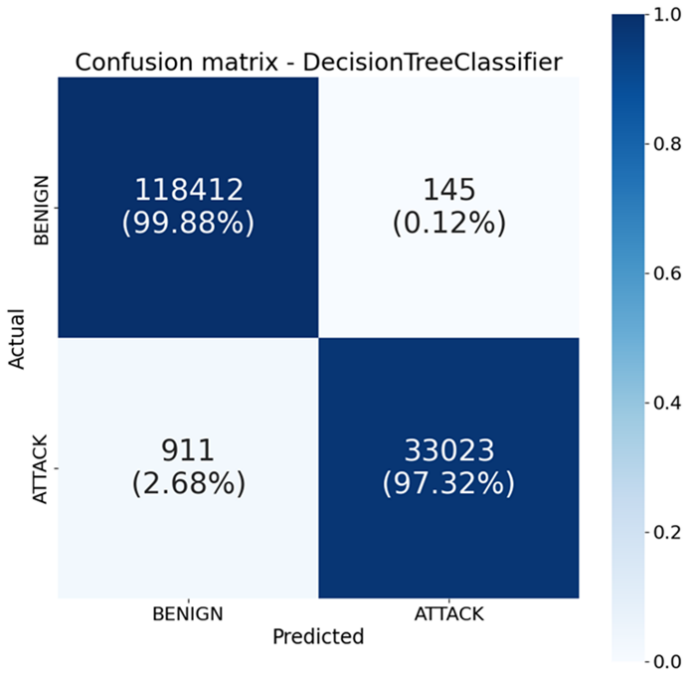

Figura 10: Matrice di confusione per il set di test della variante NetFlowMeter, max_depth=1, solo traffico del lunedì e mercoledì

Analizzando le due distribuzioni specifiche per classe della feature Bwd Packet Len Std si osserva una netta separazione. Questo suggerisce che la feature in questione è un forte discriminante tra flussi benigni e flussi relativi ad attacchi DoS.

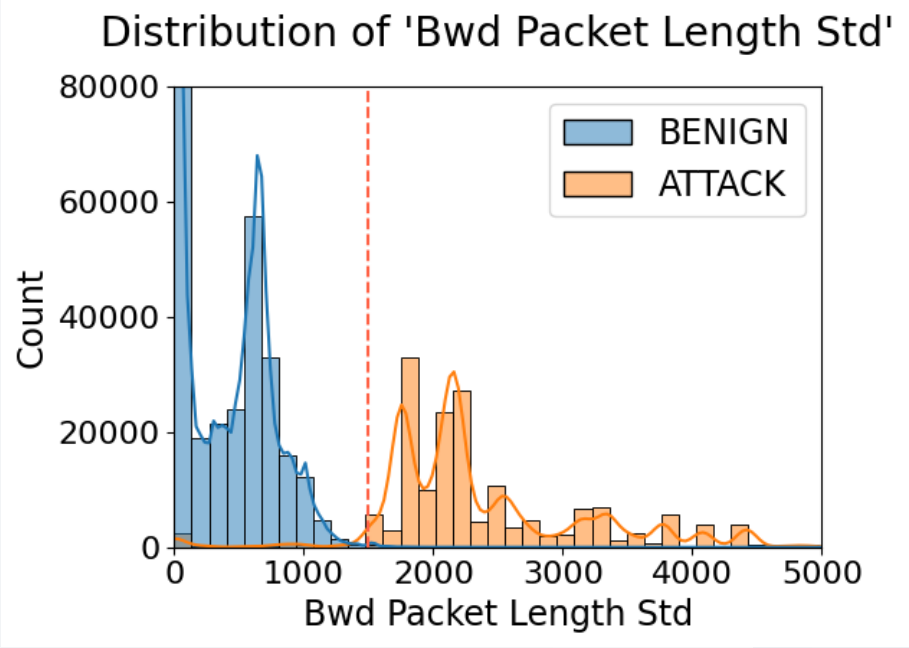

Figura 11: Distribuzioni specifiche per classe di Bwd Packet Len Std

A seguito di un'analisi approfondita del traffico del mercoledì in Wireshark, è emersa una significativa ridondanza dei dati. Si tratta di richieste GET sulla directory radice di un sito Web con query casuali, con il server che restituisce sempre la stessa pagina. Inoltre, sono stati rilevati flussi anomali con indirizzo IP di sorgente 8.6.0.1 e destinazione 8.0.6.4, con porte e protocollo pari a 0. Il filtraggio del file PCAP per questi indirizzi non ha prodotto alcun risultato. Approfondendo la questione, è emerso che l'anomalia nei dati è dovuta a pacchetti ARP interpretati erroneamente da CICFlowMeter.

4. Rilevamento di anomalie

4.1. Autoencoder

In questa sezione, si illustra una tecnica di deep learning per il rilevamento delle intrusioni che si è dimostrata generalmente efficace per dati generati utilizzando NetFlowMeter. L'approccio è semi-supervisionato e consiste nel rilevare le anomalie addestrando un modello esclusivamente su dati di traffico normali.

Il principale vantaggio di questa tecnica è che non richiede istanze di traffico malevolo per l'addestramento. È noto che le anomalie sono spesso rare, costose o addirittura impossibili da raccogliere ed etichettare. Pertanto, non è realistico aspettarsi che si possano raccogliere esempi per ogni potenziale attacco e configurazione degli strumenti esistenti. Inoltre, questa tecnica può aiutare a identificare minacce non note a priori e future.

Per implementare questo approccio, si può utilizzare una rete neurale feedforward chiamata autoencoder, caratterizzata da una struttura simmetrica a clessidra, composta da due parti principali: l'encoder, che comprime l'input in una rappresentazione a più bassa dimensionalità in un layer centrale detto bottleneck, e il decoder, che tenta di ricostruire l'input a partire dalla rappresentazione compressa.

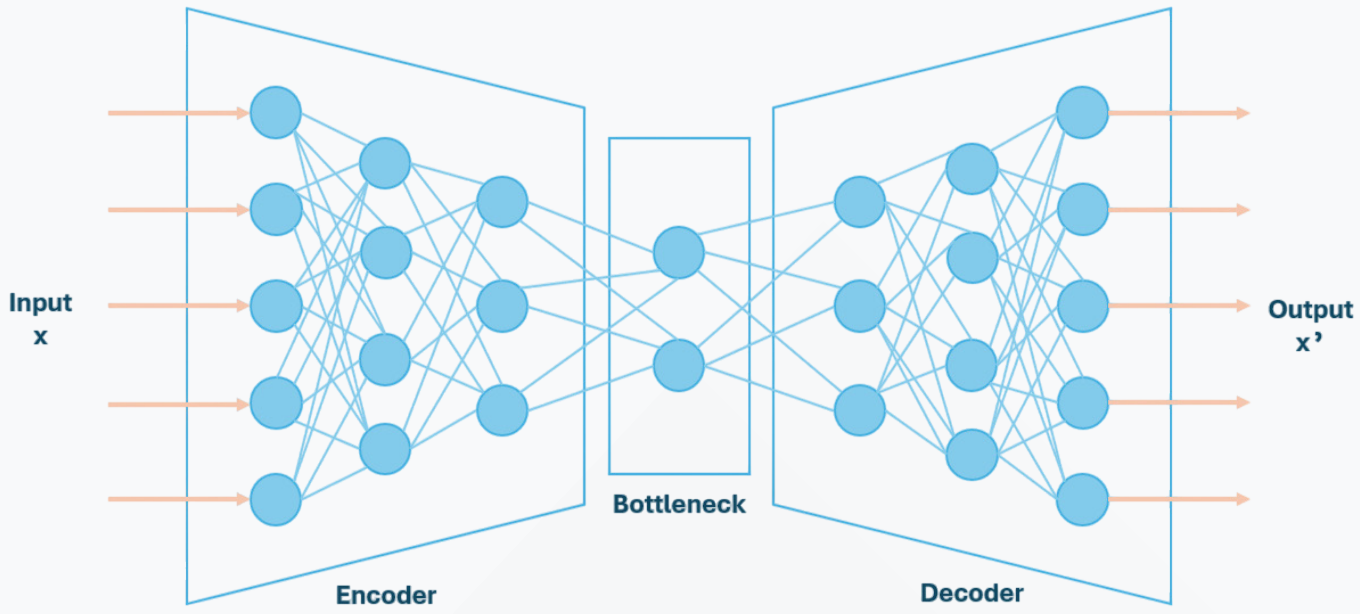

Figura 12: Struttura di un autoencoder

Si può utilizzare l'errore di ricostruzione di questo processo, tipicamente calcolato come errore quadratico medio (mean squared error, MSE), per il rilevamento delle anomalie: in virtù del fatto che l'autoencoder è stato addestrato unicamente su dati di traffico normali, sarà efficace nella codifica e ricostruzione di questi ultimi, con un conseguente errore di ricostruzione basso. D'altra parte, i dati anomali comporteranno un errore di ricostruzione più alto. È evidente la necessità di stabilire una soglia per distinguere tra errori bassi e alti, che può essere ottenuta in vari modi.

CICIDS2017 è nuovamente utilizzato come dataset di riferimento, in particolare la variante NetFlowMeter 2.0.

L'autoencoder è stato addestrato sui dati normali del lunedì suddivisi in set di addestramento (80%) e di validazione (20%), calcolando la soglia su quest'ultimo come un percentile molto alto degli errori di ricostruzione. Successivamente, il modello è stato testato su tutti gli altri giorni, mantenendoli separati per poter confrontare le prestazioni. Sono state omesse le colonne precedentemente escluse per gli alberi decisionali, e i dati sono stati scalati utilizzando MinMaxScaler di scikit-learn.

Per la costruzione del modello, sono stati impiegati i framework di machine learning TensorFlow e Keras. Per l'encoder e il decoder è stato utilizzato un singolo layer densamente connesso con 56 unità, mentre per il bottleneck, anch'esso densamente connesso, sono state utilizzate 8 unità. La funzione di loss scelta è l'errore quadratico medio. Inoltre, è stata applicata la normalizzazione dei batch per accelerare e stabilizzare l'addestramento.

Layer	Tipo	Numero di unità
Input layer	Input	86
Encoder	Dense	56
Normalizzazione dei batch	BatchNormalization	56
Funzione di attivazione	LeakyReLU	56
Bottleneck	Dense	8
Normalizzazione dei batch	BatchNormalization	8
Funzione di attivazione	LeakyReLU	8
Decoder	Dense	56
Normalizzazione dei batch	BatchNormalization	56
Funzione di attivazione	LeakyReLU	56
Output layer	Dense	86

4.2. Risultati

Di seguito si riportano i risultati ottenuti dal martedì al venerdì. Il tasso di falsi negativi (false negative rate, FNR) è basso, ma quello dei falsi positivi (false positive rate, FPR) è significativamente più alto, raggiungendo il 10% per il mercoledì.

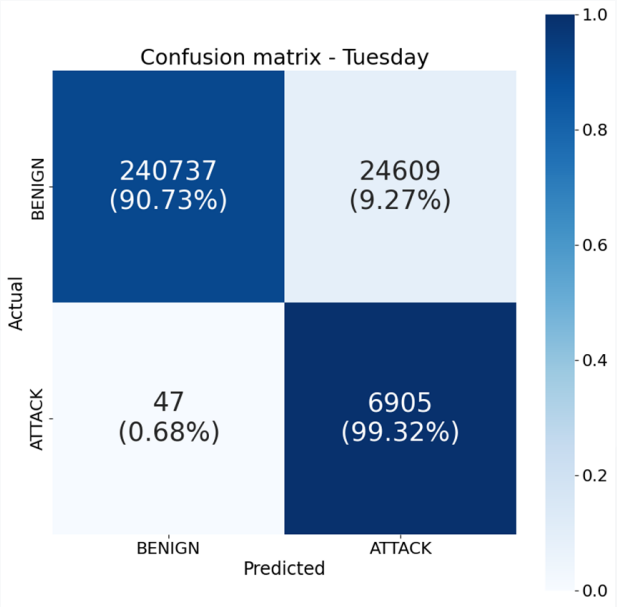

Figura 13:
Matrice di confusione per il martedì

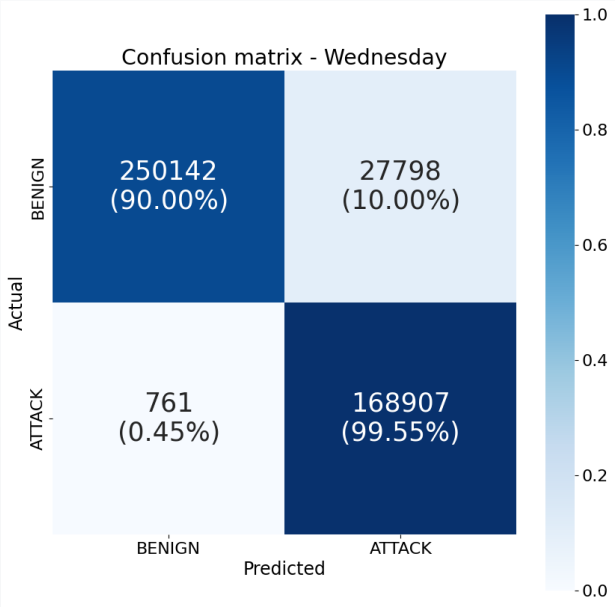

Figura 14:
Matrice di confusione per il mercoledì

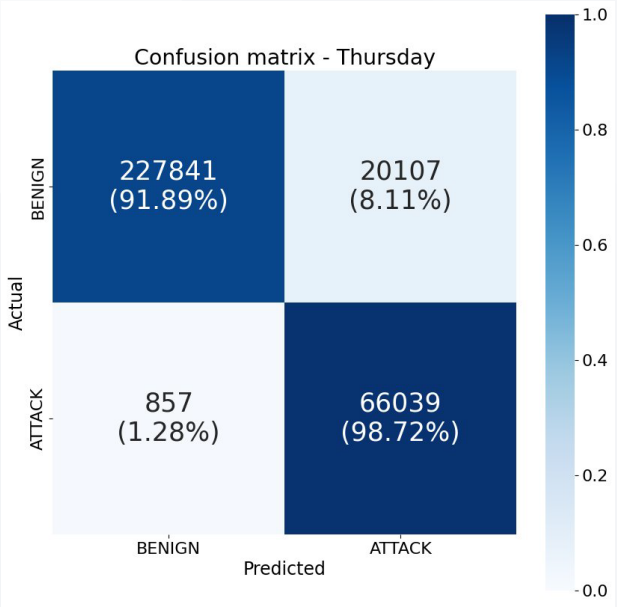

Figura 15:
Matrice di confusione per il giovedì

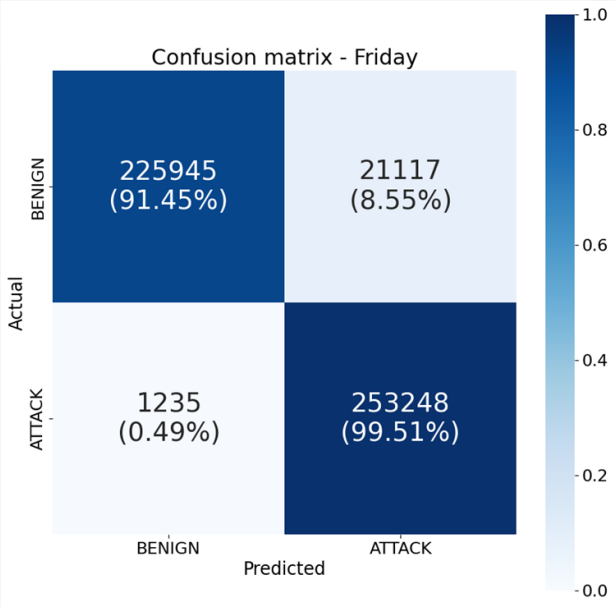

Figura 16:
Matrice di confusione per il venerdì

Come nota a margine, è possibile visualizzare gli errori di ricostruzione utilizzando un diagramma come il seguente (su scala semi-logaritmica), con l'errore di ricostruzione sull'asse y e l'indice nell'array degli errori sull'asse x. Questo può risultare utile per un'ispezione più approfondita dei risultati.

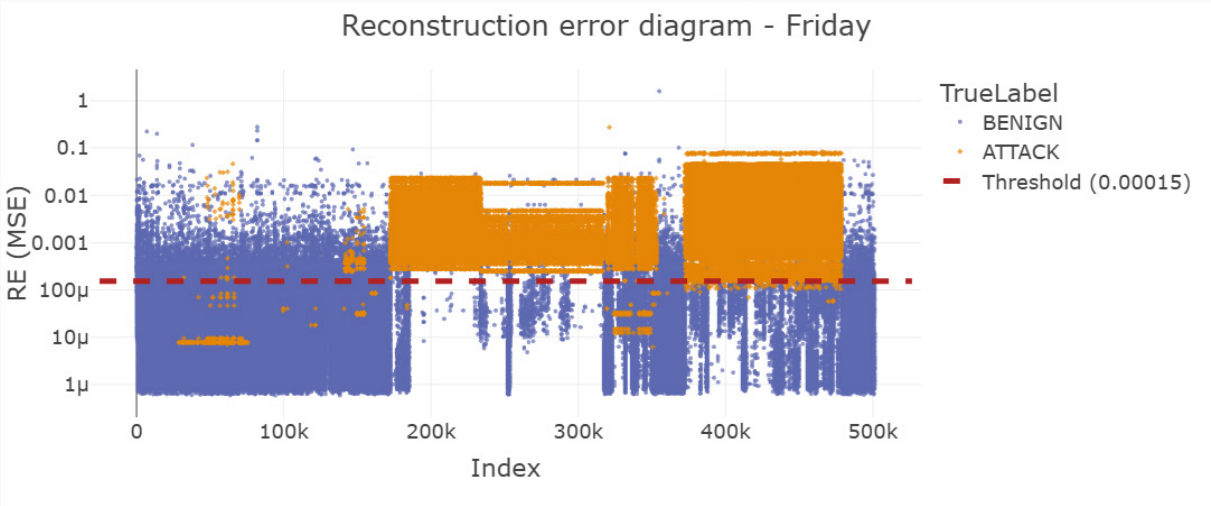

Figura 17: Esempio di diagramma dell'errore di ricostruzione

È possibile effettuare un tentativo inserendo un'unica unità nel bottleneck anziché 8. In questo caso, le prestazioni risultano deludenti, ma quelle del mercoledì rimangono in linea con quelle ottenute in precedenza. Questo risultato è attribuibile alla ridondanza presente in quella giornata di attività, come emerso in fase esplorativa.

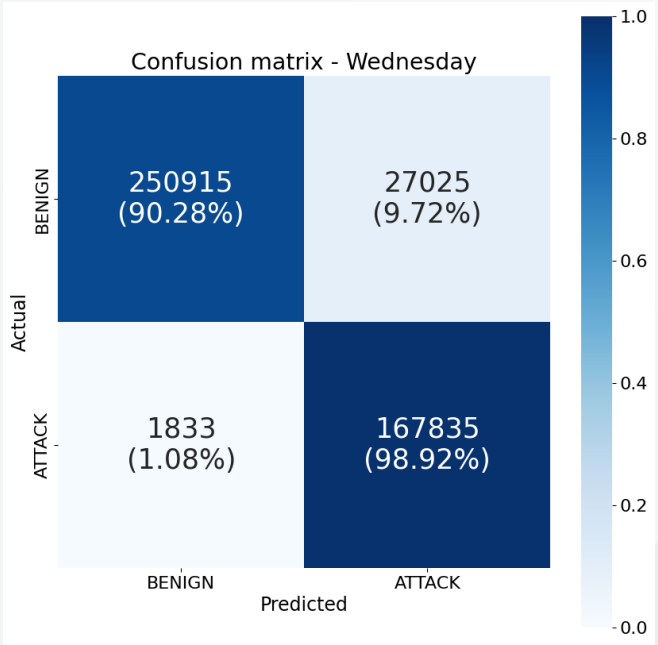

Figura 18: Matrice di confusione per il mercoledì, una sola unità nel bottleneck

Inoltre, è possibile ripristinare la logica di troncamento di CICFlowMeter che taglia i flussi ogni 2 minuti. In tal caso, le prestazioni calano significativamente il giovedì e il venerdì. Sulla base di questo risultato, sembrerebbe che non troncare i flussi sia vantaggioso per le prestazioni. Si ritiene che ciò sia dovuto al fatto che i flussi normali nello scenario senza troncamenti hanno una durata media più estesa rispetto agli attacchi, rendendo più semplice la loro distinzione.

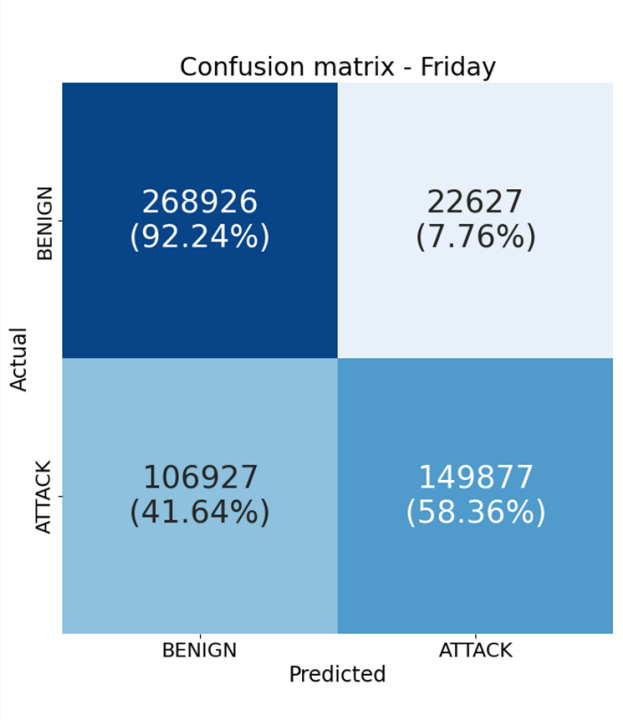

Figura 19: Matrice di confusione per il giovedì, con troncamento

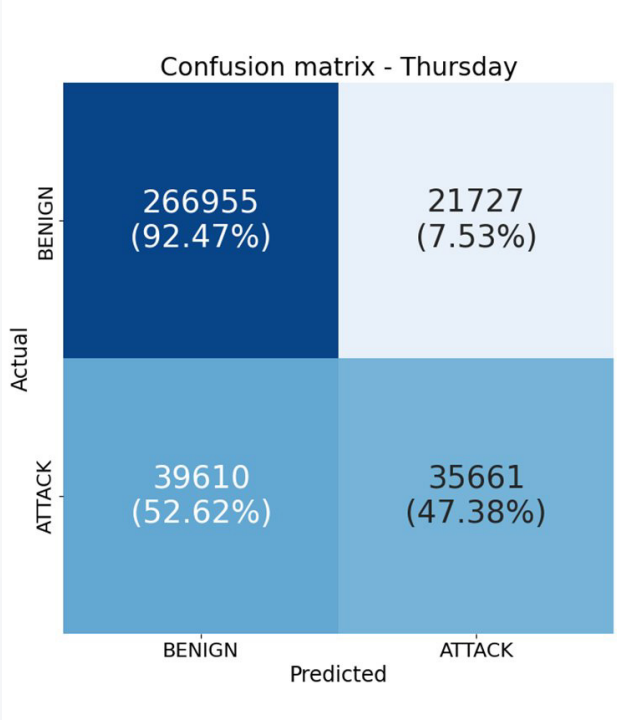

Figura 20: Matrice di confusione per il venerdì, con troncamento

5. Conclusioni

Dall'analisi dei risultati ottenuti nel processo di rilevamento delle anomalie, emerge una problematica ricorrente legata alla presenza di falsi positivi. Un elevato numero di falsi allarmi conduce all'alert fatigue, una condizione in cui l'eccessivo volume di notifiche, spesso irrilevanti, riduce la capacità degli operatori di prestare la dovuta attenzione agli allarmi effettivamente critici. Per mitigare tale rischio, il tasso di falsi positivi dovrebbe idealmente tendere a zero. Al fine di ridurre i falsi allarmi, si potrebbe:

- adottare modelli alternativi o più complessi, come gli autoencoder variational;
- prendere in considerazione metodi di apprendimento d'insieme (ensemble learning);
- utilizzare il modello in sinergia con il filtraggio basato su regole degli IDS tradizionali.

In conclusione, dall'analisi condotta emergono le seguenti indicazioni strategiche:

- non fidarsi ciecamente degli strumenti e dei dataset di terze parti, ma si consiglia di analizzare sempre le catture del traffico e le feature estratte prima di utilizzarle in applicazioni di ML;
- potrebbero essere necessarie ulteriori elaborazioni prima o dopo la classificazione basata su ML;
- non utilizzare un solo strumento e/o una sola configurazione per generare attacchi relativi a una certa tipologia, perché ciò potrebbe portare alla profilazione dello strumento o del server invece che dell'attacco vero e proprio.

Bibliografia

[1] «CICFlowMeter,» [Online]. Available: https://www.unb.ca/cic/research/applications.html#CICFlowMeter.

[2] «NetFlowMeter (GitHub),» [Online]. Available: https://github.com/DefenceTechSecurity/NetFlowMeter.

[3] «Open source release: NetFlowMeter,» [Online]. Available: https://tinextadefence.it/wp-content/uploads/2025/10/Report-Open_source_release.pdf.

[4] S. Iman, A. H. Lashkari e A. A. Ghorbani, «Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,» in 4th International Conference on Information Systems Security and Privacy (ICISSP), Funchal, Madeira, Portugal, 2018.

[5] «Intrusion detection evaluation dataset (CIC-IDS2017),» [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html.

[6] L. Liu, G. Engelen, T. Lynar, D. Essam e W. Joosen, «Error Prevalence in NIDS datasets: A Case Study on CIC-IDS-2017 and CSE-CIC-IDS-2018,» in 2022 IEEE Conference on Communications and Network Security (CNS), Austin, TX, USA, 2022.

[7] «Index of /CNS2022/Datasets,» [Online]. Available: https://intrusion-detection.distrinet-research.be/CNS2022/Datasets/.

[8] «Improved CIC-IDS 2017 Documentation,» [Online]. Available: https://intrusion-detection.distrinet-research.be/CNS2022/CICIDS2017.html.

[9] «Decision Trees,» [Online]. Available: https://scikit-learn.org/stable/modules/tree.html.

[10] «Permutation feature importance,» [Online]. Available: https://scikit-learn.org/stable/modules/permutation_importance.html.

tinexta
defence

Defence Tech | Next |
Donexit | Foramil | Innodesi

Via Giacomo Peroni, 452 – 00131 Roma
tel. 06.45752720 – info@defencetech.it
www.tinextadefence.it

#TinextaDefenceBusiness