Tecniche di Meta-Learning per la progettazione di un sistema di rilevamento delle intrusioni multi-livello

Tesi di Laurea Magistrale in Ingegneria Informatica

Irene Mazzamuto

Relatore: Ing. Pierluca Ferraro

Correlatore: Ing. Vincenzo Agate

TECNICHE DI META-LEARNING PER LA PROGETTAZIONE DI UN SISTEMA DI RILEVAMENTO DELLE INTRUSIONI MULTI-LIVELLO

Tesi di Laurea di
Dott.sa Irene Mazzamuto

Relatore:
Ing. Pierluca Ferraro

Correlatore:
Ing. Vincenzo Agate

Abstract

Gli Intrusion Detection Systems sono strumenti di rilevamento delle anomalie di rete utilizzati nell'ambito della Network Security al fine di monitorare continuamente il traffico alla ricerca di attività sospette o dannose, come tentativi di accesso non autorizzato, malware o attacchi distribuiti. Gli IDS si differenziano principalmente in base ai principi che utilizzano per identificare le minacce durante l'analisi del traffico. Le macrocategorie di IDS attualmente più popolari basano il riconoscimento degli attacchi su insiemi di *firme di attacchi* conosciuti, limitando però la loro capacità di generalizzazione, o sull'*apprendimento automatico*, per poter analizzare i dati ed estrarre dei pattern di rilevamento senza necessità di intervento umano.

Questa tesi si focalizza sugli IDS basati sul Machine Learning prendendo in esame le teorie e gli strumenti necessari a realizzare un Intrusion Detection System gerarchico in grado di rilevare le minacce nella rete. Particolare enfasi è stata data allo studio degli approcci di Meta-Learning, i quali sfruttano più modelli semplici di machine learning per poi aggregarne i risultati concorrendo alla realizzazione di un unico output. Le tecniche maggiormente adoperate in questo contesto, fanno parte della famiglia degli algoritmi di Ensemble Learning, di cui saranno presentati diversi studi e alcune delle loro implementazioni, sfruttate poi per la realizzazione del suddetto IDS.

Per migliorare la capacità di generalizzazione dell'Intrusion Detection System basato sull'Ensemble Learning, sono state esplorate diverse tecniche di trasformazione, visualizzazione dei dati e clustering, al fine di identificare le caratteristiche che contribuiscono maggiormente ad una corretta classificazione degli attacchi.

Alla fine dello studio, sono stati presentati i benchmark dell'IDS realizzato sul dataset CIC-IDS2018, un dataset sintetico contenente istanze sia di traffico benigno sia relative a 14 differenti attacchi.

«E c'è un destino che si chiama "destinazione"
Ma non ci arrivi mai se provi a andar di corsa»

Dentista Croazia - Pinguini Tattici Nucleari

Contents

List of Figures

Acronimi

CSV	Comma Separated Values
DBMS	Database Management System
DDoS	Distributed Denial of Service
DoS	Denial of Service
DVWA	Damn Vulnerable Web Application
FIFO	First-In First-Out
FTP	File Transfer Protocol
GBDT	Gradient Boosted Decision Trees
HOIC	High Orbit Ion Cannon
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IQR	Interquantile Range
LGBM	Light Gradient Boosting Machines
LOIC	Low Orbit Ion Cannon
MIT	Massachusetts Institute of Technology
ML	Machine Learning
NFL	No Free Lunch
PCA	Principal Component Analysis
RFC	Request for Comments
SMOTE	Syntetic Minority Oversampling Tecnique
SMTP	Simple Mail Transfer Protocol
SNE	Stochastic Neighbor Embedding
SSH	Secure Shell
t-SNE	t-distributed Stochastic Neighbor Embedding
VPS	Virtual Private Server

WCSS	Within-cluster sum-of-squares
WWW	World Wide Web
XGB	eXtreme Gradient Boosting
XSS	Cross Site Scripting

Chapter 1

Introduzione

> Le aziende produttrici di software dovrebbero assumersi maggiori responsabilità per le falle di sicurezza, soprattutto nei browser e nei client di posta elettronica. Ci sono alcune cose semplici che l'industria dovrebbe fare adesso per sistemare le cose, e non so perché non siano ancora state fatte.
>
> *Tim Berners-Lee* - Ideatore World Wide Web

È il 1988 quando Robert Morris, studente della Cornell University, deciso a vagliare le dimensioni della rete Internet del tempo, scrive quello che oggi è conosciuto come *Morris Worm*, la prima registrazione storica di un malware che si diffonde attraverso Internet. All'epoca dei fatti, la rete Internet era molto diversa: veniva utilizzata principalmente per scopi di ricerca, scientifici e militari. Una nota rilasciata nel Gennaio 1992 destinata alla Internet Community dei tempi, l'RFC1296 [1], riporta infatti che nell'Ottobre del 1988 meno di 60mila calcolatori erano connessi alla rete.

Il Morris Worm, lanciato da un calcolatore del MIT, infettò un sistema dopo l'altro fino ad essere arrestato, secondo le stime, a 6000 calcolatori infetti[1], circa il **10% dell'Internet conosciuto**.

Era implementato in maniera estremamente semplice: riusciva ad installarsi negli host sfruttando quattro vulnerabilità dei sistemi UNIX, inviava un messaggio all'host che ne aveva iniziato la diffusione affinchè incrementasse il conteggio dei calcolatori connessi e continuava la sua replicazione. Ogni volta che un calcolatore veniva spento, si perdeva totalmente traccia dell'installazione, cancellando il worm dal sistema. Un effetto collaterale di questa implementazione fu che il worm agì come DDoS (Distributed Denial of Service)[2] in quanto i calcolatori infetti divennero inutilizzabili in poche ore dall'infezione, causando un'isteria di massa, totalmente inattesa.

[1] Kaspersky Lab: *Morris Worm Turns 25* `https://www.kaspersky.com/blog/morris-worm-turns-25/3065/`

[2] Security Encyclopedia: Morris Worm`https://www.hypr.com/security-encyclopedia/morris-worm`

Due anni dopo, nel 1991, gli Stati Uniti D'America[3] deliberarono contro Morris segnando una pietra miliare nella storia della **Cybersecurity**.

Nei decenni successivi a questo avvenimento, complice l'invenzione del World Wide Web ad opera di Tim Berners-Lee, Internet subisce una rapida crescita: le informazioni sono sempre più accessibili grazie all'avvento dei siti web. Nel Dicembre 2005 più di 1 miliardo di utenti è connesso ad Internet[4]. Ma la vera rivoluzione si ha nel Gennaio 2007, quando a San Francisco, Steve Jobs, l'allora amministratore delegato di Apple, presenta l'iPhone[5]: *"un rivoluzionario dispositivo di comunicazione Internet con email di livello desktop, web browsing, funzionalità di ricerca e mappe"*. Da quel momento sempre più persone poterono accedere ad Internet attraverso i dispositivi mobili che sarebbero entrati nel mercato fino a portarci, nel Luglio 2022, al **69% della popolazione mondiale connessa ad Internet**.

La nascita del World Wide Web fu un tassello fondamentale per l'evoluzione di Internet e di quello che negli anni prese il nome di **Web 1.0**: una versione *statica* del web, dove gli utenti potevano consultare le informazioni liberamente ma senza una reale interazione. Ma le cose erano destinate a cambiare, infatti nel 2004 viene coniato il termine **Web 2.0** [2]: una versione del web "incentrata sulle persone", dove la parola chiave è **interazione**, non più un web statico dove si potevano semplicemente consultare le informazioni, bensì una versione del web dove le persone possono interagire con i contenuti in tempo reale. Al giorno d'oggi Internet non è più semplicemente una rete di calcolatori come lo era negli anni '90, si è evoluto tanto da diventare parte della vita quotidiana. Il fenomeno della digitalizzazione ha semplificato il modo col quale le persone si rapportano con gli enti di cui necessitano: oggi le banche, gli ospedali, le attività commerciali, i servizi pubblici, la gestione dell'identità, persino il lavoro d'ufficio, è su Internet.

Ma la digitalizzazione non si è limitata a portare su Internet solo i lati positivi della vita quotidiana. Parallelamente allo sviluppo di queste applicazioni di vitale importanza sono cresciute anche le minacce e i crimini collegati alla rete [3] e per garantire la sicurezza e la stabilità di Internet, la **Cybersecurity** ricopre un ruolo fondamentale della vita di tutti i giorni.

Questa tesi si focalizzerà sullo studio degli Intrusion Detection System, in particolare quelli basati sul Machine Learning, al fine di realizzare un IDS multi-livello basato su tecniche di clustering e di Ensemble Learning. L'architettura multi-livello è stata adottata al fine di effettuare una discriminazione delle minacce facendo uso del clustering dei dati, il quale ha consentito di discriminare gli attacchi raggruppandoli sulla base delle caratteristiche loro comuni. Le tecniche di Ensemble Learning che si andranno ad esaminare si basano sul concetto di Meta-Learning, un approccio all'apprendimento automatico che mira a sviluppare classificatori più robusti attraverso la combinazione di più modelli mediante l'uso di un meta-livello. Questo consente di ottenere prestazioni migliori rispetto a quelle ottenute con un singolo modello poiché i modelli realizzati si bilanciano vicendevolmente apprendendo diversi aspetti dei dati che concorrono ad ottenere classificazioni più precise.

[3]US vs Morris: `https://scholar.google.com/scholar_case?case=5513862414516396 68&hl=it&as_sdt=0`

[4]Internet World Stats: `https://www.internetworldstats.com/emarketing.htm`

[5]Apple reinventa il telefono con l'iPhone:`https://www.apple.com/it/newsroom/2007/01/09Apple-Reinvents-the-Phone-with-iPhone/`

Nel Cap. 2 saranno descritte le diverse tipologie di IDS presenti allo stato dell'arte, ponendo il focus sugli IDS basati sul machine learning, per poi introdurre l'utilizzo del Meta-learning per la realizzazione degli IDS. Nella Sez. 2.3 saranno presentate le tecniche di Ensemble Learning che costituiscono lo stato dell'arte, comprese le famiglie degli algoritmi di Bagging e Boosting delle quali sono state usate diverse implementazioni per la realizzazione dell'IDS.

Nel Cap. 3 sarà proposta l'architettura per la realizzazione di un IDS multi-livello e gli algoritmi utilizzati per la realizzazione dei singoli livelli. Inoltre, nella Sez. 3.3, saranno presentati gli algoritmi di clustering e riduzione della dimensionalità, usati per migliorare le performance dell'IDS proposto.

Nel Cap. 4 sarà preso in esame il dataset scelto per la realizzazione dell'IDS, ovvero il *CSE-CIC-IDS2018*, un dataset sintetico che presenta al suo interno 14 diversi attacchi di cui sarà evidenziato il funzionamento per sommi capi.

Nel Cap. 5 saranno riportate le fasi principali di realizzazione dell'IDS, insieme ai risultati ottenuti. In particolare, oltre all'analisi dei dati, saranno evidenziate le tecniche utilizzate per il clustering mettendo in evidenza la natura intrinseca dei dati e la loro distribuzione spaziale, terminando con l'assemblaggio dell'IDS vero e proprio.

Chapter 2

Osservando la rete: come prevenire un'intrusione

> Il fatto è che lei guarda ma non osserva; qui sta la differenza.
>
> *Sir Arthur Conan Doyle* - Sherlock Holmes

La rete è esposta ogni giorno ad un elevato numero di attacchi che ne minacciano la sicurezza: secondo alcune statistiche rilasciate dall'ANSA (Agenzia Nazionale Stampa Associata)[1] tra Luglio e Settembre 2022, un'azienda viene colpita da minacce informatiche circa 1130 volte a settimana. Gli attacchi informatici sono in continua evoluzione, ne sono un esempio gli *zero-day attacks*, ovvero attacchi che sfruttano delle vulnerabilità non ancora note e per i quali non è stata rilasciata una patch, ma possiamo anche prendere in considerazione attacchi più sofisticati come i *supply-chain attacks*, ovvero attacchi il cui bersaglio viene colpito indirettamente attraverso l'attacco ad uno dei servizi di cui fa uso.

È possibile generalizzare le tipologie di attacchi condotti tramite la rete in due macro categorie: attivi e passivi.

Negli attacchi attivi, uno degli scopi principali è in genere la manomissione del sistema bersaglio con un determinato scopo. Tra gli attacchi attivi è possibile prendere in esame: gli attacchi *DoS/DDoS*, che mirano a negare la fruizione di un servizio in modo aggressivo ai legittimi utenti; gli attacchi di *Phishing*, che mirano a carpire informazioni di un utente truffandolo; gli attacchi di *Defacing*, che mirano a scambiare l'interfaccia di un sistema con un'interfaccia totalmente differente al fine di mostrare che il sistema è stato attaccato ed è vittima dell'attaccante.

Negli attacchi passivi invece, lo scopo principale è quello di accedere alle informazioni trasmesse tramite la rete in maniera silente ed oculata: sono infatti i più difficili da rilevare dal momento che non comportano alcuna alterazione del normale flusso del traffico di rete. Un esempio di attacco passivo consiste nello *Sniffing*, ovvero nell'ascolto e nella raccolta del traffico al fine di rubarne il contenuto.

[1] Sicurezza, crescono in un anno del 28% gli attacchi informatici: https://www.ansa.it/sito/notizie/tecnologia/hitech/2022/11/02/sicurezza-crescono-in-un-anno-del-28-attacchi-informatici_ab483480-4192-47e2-a7af-51402c2ba44f.html

La Cybersecurity ricopre un ruolo fondamentale nel contrasto degli attacchi alle reti e il suo progresso ha consentito, negli anni, lo sviluppo di diverse misure, di natura sia hardware che software, che, installate e/o usate all'interno di una rete, consentono il rilevamento, la prevenzione, il blocco o semplicemente l'analisi del traffico di rete malevolo.

Questo studio focalizzerà la sua analisi su uno degli strumenti di rilevamento degli attacchi: l'**Intrusion Detection System**.

2.1 Intrusion Detection System

L'**IDS** è un sistema passivo di monitoraggio (hardware o software) che può essere installato in diversi punti della rete (in funzione del traffico che si vuole monitorare), che ha il compito di rilevare e notificare eventuali minacce, in funzione della configurazione che gli è stata impartita dagli amministratori di rete. E' possibile distinguere gli IDS hardware (ad esempio installati a fianco degli switch di rete) *network-based*, il cui compito potrebbe essere il monitoraggio del traffico dell'intera rete, ma senza alcun intervento in caso di rilevamento, e gli IDS software installati a livello di host o *host-based*, il cui scopo è il rilevamento delle violazioni di un sistema host. Gli IDS possono anche essere accoppiati al funzionamento di altri dispositivi di rete per fornire un ulteriore livello di sicurezza: ad esempio un IDS potrebbe essere installato in sequenza ad un firewall di rete, così da analizzare il traffico uscente da esso e identificare eventuali minacce che non sono state bloccate.

Nel caso in cui gli IDS siano dotati di meccanismi da azionare post-rilevamento che vanno al di là della semplice notifica, questi prendono il nome di *Intrusion Prevention Systems* (IPS), ovvero sistemi col compito di rilevare e successivamente contrastare eventuali minacce. Anche questi possono essere utilizzati in concomitanza di altre apparecchiature di rete per una maggiore sicurezza.

Gli IDS si distinguono in *signature-based* e *anomaly-based* sulla base della strategia adottata per il rilevamento delle intrusioni [4]:

- i sistemi *signature-based* rilevano un'intrusione solo nel caso in cui viene riscontrato un match con la firma di uno degli attacchi noti: ciò comporta che eventuali attacchi non noti non verranno rilevati;
- i sistemi *anomaly-based* rilevano un'intrusione solo nel caso in cui viene riscontrato un comportamento che si discosta dal modello di comportamento "normale" del sistema.

Il rilevamento basato sulle anomalie è attualmente l'approccio principale nel campo del rilevamento delle intrusioni.

2.1.1 Tecniche di rilevamento delle intrusioni

Esistono molteplici tecniche di rilevamento delle intrusioni per la realizzazione di IDS, tuttavia è possibile distinguere tre macro categorie [4]: *statistical-based*, *knowledge-based* e *machine learning-based*.

2.1.1A Statistical-based

Le tecniche di rilevamento delle intrusioni basate sulla statistica sfruttano delle metriche statistiche, quali per esempio media e varianza, per costruire un profilo comportamentale del normale traffico di rete. Questo verrà poi confrontato con il comportamento del traffico di rete catturato, e da tale confronto viene restituito un grado di irregolarità tanto più grande quanto più i due comportamenti si discostano. Se questo grado supera una certa soglia fissata, il traffico di rete catturato viene considerato anomalo.

Il lato positivo di questo approccio è la totale indipendenza da qualsivoglia conoscenza a priori della rete. Di contro, per via della sua natura praticamente statica, ogni comportamento differente da quello rilevato in fase iniziale, è da considerarsi anomalo, non consentendo una naturale evoluzione della rete, a meno di un ri-addestramento periodico e continuo.

2.1.1B Knowledge-based

Le tecniche di rilevamento delle intrusioni basate sulla conoscenza sfruttano dei set di regole installati appositamente dagli amministratori di sistema. Ciò implica un grande costo iniziale dovuto alla necessità di analizzare tutte le implicazioni relative alla gestione della rete e ogni comportamento al di fuori del set di regole prefissato sarà considerato anomalo.

Il lato positivo di questo approccio è la robustezza derivante dal fatto che, chi imposta i parametri di classificazione, è un esperto in materia, mentre i principali svantaggi sono legati ai costi necessari per sostenere il livello di specializzazione tale da saper costruire e mantenere nel tempo un set di regole adeguato.

2.1.1C Machine learning-based

La natura dei sistemi di rilevamento delle intrusioni basati sul machine learning coincide con quella dei sistemi basati sulle statistiche. La differenza fondamentale tra i due consiste nell'implementazione del sistema stesso: nel caso dei sistemi basati sulle statistiche, i dati relativamente alla rete sono "statici", mentre nel caso di modelli basati sul machine learning, i pattern e le associazioni tra i dati sono inferite automaticamente dal modello.

Il lato positivo di questo approccio è la totale indipendenza del modello rispetto alla rete ed il fatto che non necessiti di alcun intervento umano per la manutenzione ed il ri-addestramento. Tuttavia, questa soluzione può essere economicamente onerosa da mantenere.

2.2 Approcci di Meta-Learning per IDS Machine Learning-Based

Negli anni diversi studi si sono focalizzati sull'utilizzo degli algoritmi di machine learning per la realizzazione degli IDS, considerando il problema del rilevamento delle intrusioni come un problema di classificazione del traffico benigno e degli attacchi.

Un problema di machine learning può essere espresso genericamente come un problema di ottimizzazione di una funzione di apprendimento f all'interno di uno specifico dominio θ. In questo scenario di ottimizzazione si pone il "No Free Lunch Theorem" [5], il quale afferma l'impossibilità di trovare una funzione f *general-purpose* in grado di fornire una strategia di ottimizzazione universale, rendendo dunque necessario, per ogni dominio applicativo, lo studio di una funzione f specializzata nella risoluzione del problema. Una conseguenza del NFL comporta che senza una conoscenza a priori del dominio θ (ovvero del problema in tutte le sue sfaccettature), tutte le tecniche che si vogliono utilizzare, portino alle stesse performance.

2.2.1 Meta-Learning

Un approccio per gestire il problema introdotto dal NFL, è l'utilizzo del **Meta-Learning** (o Meta-Apprendimento). L'idea alla base del Meta-Learning è quella di sfruttare la meta-conoscenza che si ha sui modelli di machine learning per selezionare più efficacemente la soluzione da adottare sul dominio selezionato.

Il Meta-Learning è stato concettualmente introdotto negli anni '80 [6], tuttavia vede la sua applicazione pratica molti anni più avanti [7] grazie allo sviluppo tecnologico che ha fornito sempre più potenza computazionale per le tecniche di apprendimento. La formulazione del meta-learning prevede due livelli di applicazione: *meta-feature* e *meta-livelli di apprendimento*.

2.2.1A Meta-Feature

Le **Meta-Feature** sono delle feature derivate dai dati originali che possono essere distinte in 5 categorie in funzione delle loro peculiarità:

- *feature semplici*: ovvero provenienti dal dataset originale;
- *feature statistiche*: descrivono la natura statistica dei dati;
- *feature informative/teoriche*: calcolano esplicitamente l'apporto informativo dei dati nel dataset;
- *feature regionali*: definiscono delle aree del dataset in cui i *meta-livelli* apprendono meglio;
- *Model-based meta-feature*: feature estratte successivamente alla fase di apprendimento e specifiche dei modelli utilizzati.

2.2.1B Meta-livelli di Apprendimento

Un'altra sfaccettatura del Meta-Learning riguarda la combinazione di diversi modelli di Machine Learning: in particolare, si parla di **meta-livelli di apprendimento** quando si fa riferimento alla possibilità di un classificatore di sfruttare più modelli interni per effettuare la classificazione. Un meta-learner è dunque un modello composto da diversi *base-learners* o *weak-learners* che producono degli output intermedi che concorrono alla classificazione finale adoperata dal *meta-livello*.

Sempre più algoritmi implementano concettualmente l'idea dei meta-learners, combinando più modelli semplici ed aggregandone i risultati attraverso l'uso di un meta-livello: questa famiglia di modelli prende il nome di modelli di **Ensemble Learning**.

2.3 Meta-Learning: Ensemble-Learning

Nelle tecniche di Ensemble Learning che saranno di seguito presentate è possibile osservare il comportamento dei *base learners* all'interno dei *meta-learners* e come i *meta-livelli* aggregano i risultati.

Risulta inoltre particolarmente interessante come studi recenti [8] [9] [10] [11], focalizzati su IDS realizzati attraverso tecniche di *ensemble learning*, abbiano dimostrato particolare efficacia in tal contesto, anche rispetto alle comuni tecniche di machine learning.

2.3.1 Bagging

La prima tecnica che sarà presa in esame prende il nome di **Bagging**, ovvero *Bootstrap Aggregating* e consiste in una tecnica di ensemble learning che combina diverse copie di uno stesso *base-learner* addestrate su sotto-insiemi differenti estratti dallo stesso dataset. Questa tecnica venne presentata da *Leo Breiman* [12] nel 1996.

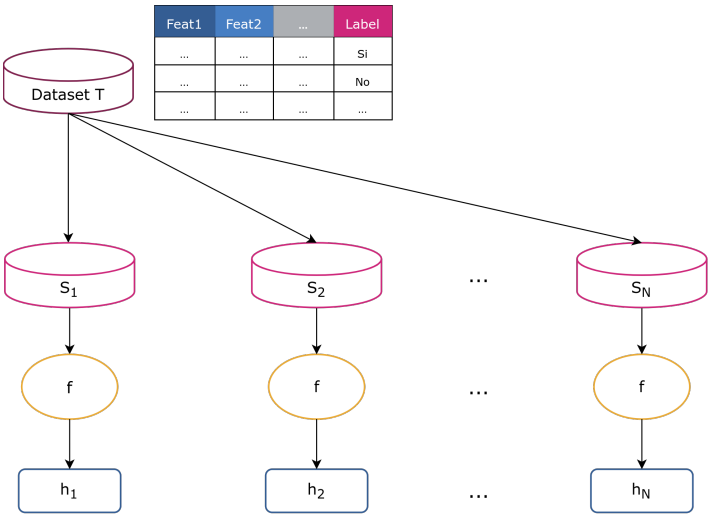

Figure 2.1: *Bagging*

Un modello di Bagging viene addestrato partendo da un dataset di training T e un algoritmo di apprendimento f. Ogni i-esimo *base-learner* consiste in una particolare istanza dell'algoritmo f, addestrato sull'i-esimo sottoinsieme del dataset di training. Si ottiene così un insieme di modelli $\{h_1, ..., h_N\}$ (Fig. 2.1).

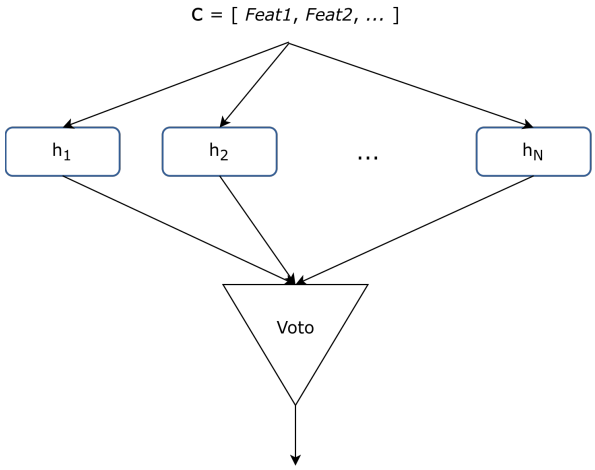

Figure 2.2: *Bagging: classificazione di un nuovo campione*

Preso un nuovo campione c, questo verrà dato in input ai singoli modelli h_i, i quali restituiranno N classificazioni differenti. Il risultato finale sarà dato dall'etichetta più votata (Fig. 2.2).

2.3.2 Boosting

Per **Boosting** si intende una tecnica di ensemble learning che consiste nel combinare più *weak learners* al fine di ottenere un modello più preciso. Questa tecnica venne proposta da *Robert E. Schapire* [13] nel 1990.

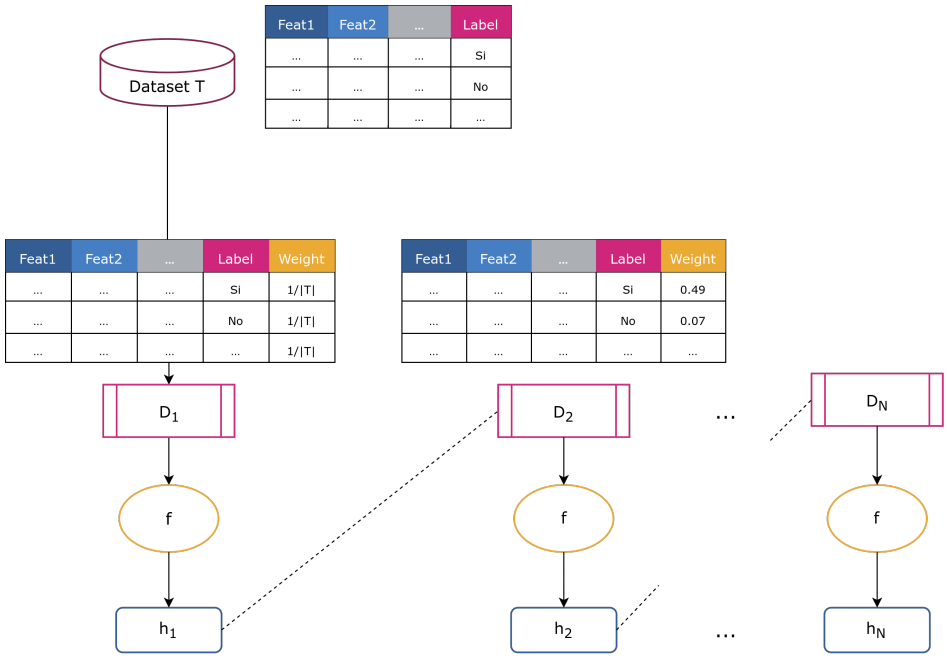

Figure 2.3: *Boosting*

Il Boosting considera una distribuzione iniziale uniforme D_1 sul dataset di training T, dove ad ogni istanza è assegnato un peso costante pari alla probabilità $\frac{1}{|T|}$ di essere estratta dal dataset (Fig. 2.3). Il primo learner, addestrato sulla distribuzione D_1, restituisce il modello h_1.

Ad ogni iterazione i pesi relativi alle istanze classificate erroneamente vengono incrementati in modo tale che il learner successivo si focalizzi maggiormente su queste istanze.

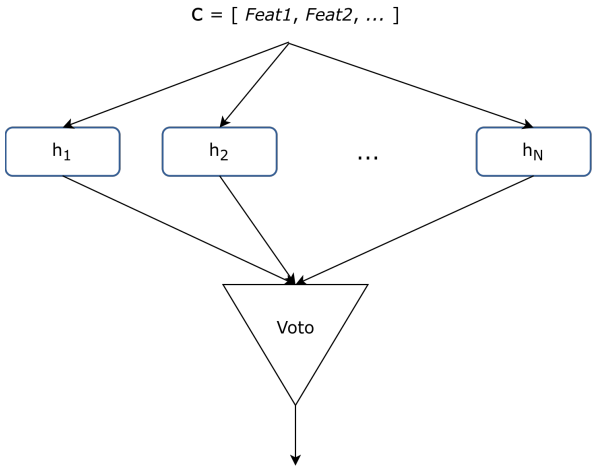

Figure 2.4: *Boosting: classificazione di un nuovo campione*

Come nel Bagging, dato un nuovo campione c, questo verrà dato in input ai singoli modelli h_i e l'etichetta restituita sarà quella più votata (Fig. 2.4).

2.3.3 Stacking

Per **Stacking** si intende una tecnica di ensemble learning che consiste nel combinare più *base-learners* diversi tra loro al fine di ottenere un modello più preciso. Questa tecnica venne proposta da *David Wolpert* [14] nel 1992.

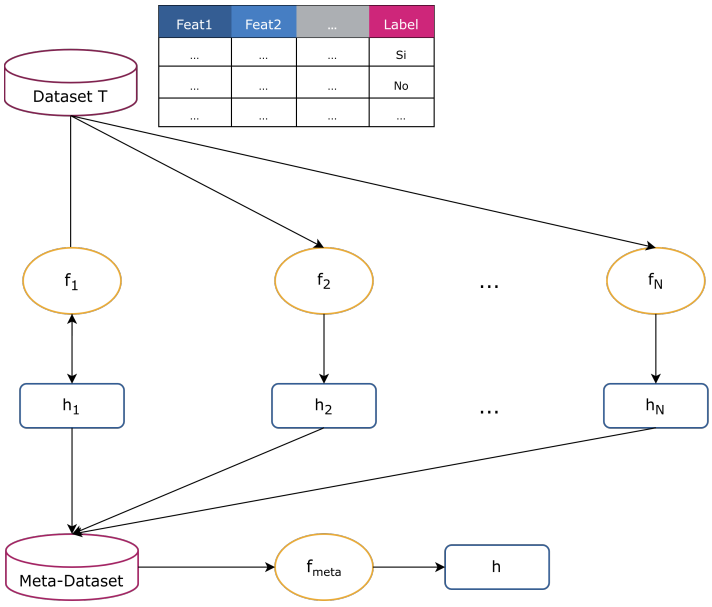

Figure 2.5: *Stacking*

Un modello di Stacking considera N algoritmi di apprendimento $t_1, \ldots, t_N$, ognuno dei quali viene addestrato sull'intero dataset di training T producendo i modelli $h_1, \ldots, h_N$. Le predizioni dei modelli costruiscono un nuovo dataset $T2$, rappresentante

un Meta-Dataset, sul quale verrà addestrato l'algoritmo f_{meta} che produrrà il modello h (Fig. 2.5).

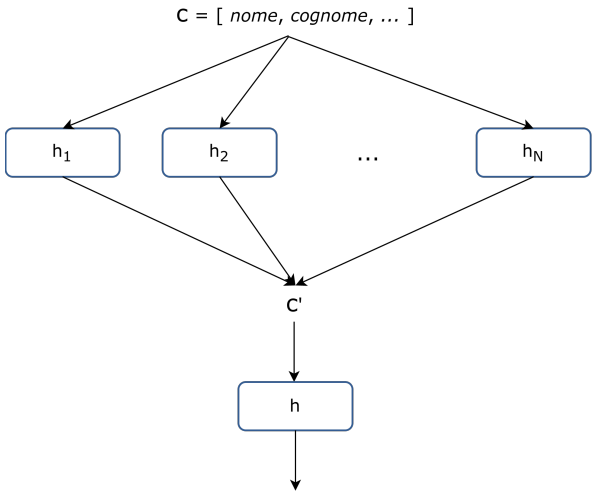

Figure 2.6: *Stacking: classificazione di un nuovo campione*

Dato un nuovo campione c, questo verrà processato in parallelo dai vari modelli h_i, portando così alla generazione del meta-dato $c' = \{h_1(c), h_2(c), ..., h_N(c)\}$ che verrà dato in input al meta-modello h, il quale produrrà il risultato (Fig. 2.6).

2.3.4 Cascading

Per **Cascading** si intende una tecnica di ensemble learning che consiste, come nel caso del Boosting, nel combinare più *weak learners* al fine di ottenere un modello più preciso, con la differenza che in questo caso i *weak learners* si basano su algoritmi di apprendimento differenti. Questa tecnica venne proposta da *Alpaydin Ethem* e *Kaynak Cenk* [14] nel 1992.

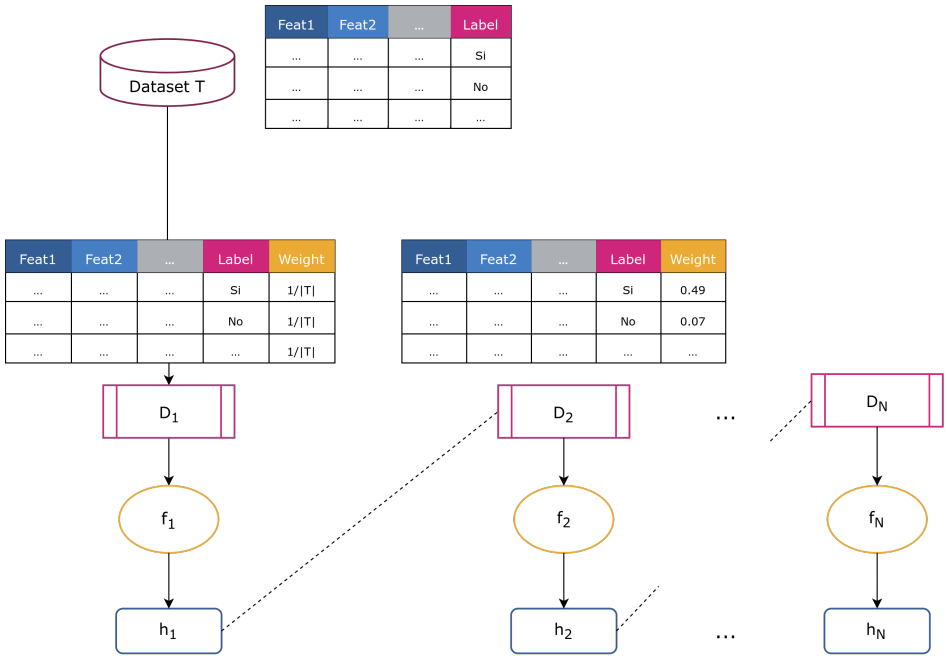

Figure 2.7: *Cascading*

Un modello di Cascading considera una distribuzione iniziale uniforme D sul dataset di training T, dove ad ogni istanza è assegnato un peso costante pari alla probabilità $\frac{1}{|T|}$ di essere estratta dal dataset (Fig. 2.7). Il primo *learner* viene addestrato sulla distribuzione D_1, restituendo il modello h_1. Ad ogni iterazione i pesi relativi alle istanze classificate erroneamente vengono incrementati in modo tale che il learner successivo si focalizzi maggiormente su queste istanze.

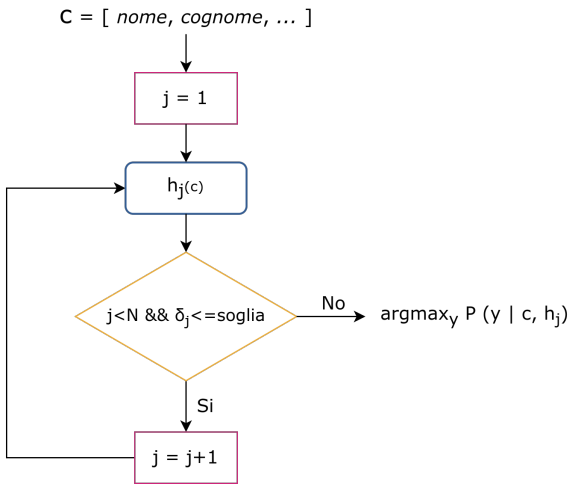

Figure 2.8: *Cascading: classificazione di un nuovo campione*

Dato un nuovo campione c, questo verrà dato in input ai modelli h_i che, diversamente dal Boosting, non sono combinati secondo uno schema di votazione ma sono posti in sequenza. Ad ogni modello è associata una funzione δ_i per il calcolo della confidenza corrispondente. In particolare, essendo

$$g_{ji}(x) = P(Cl_i|\mathbf{x}, f_j)$$

la probabilità a posteriori, stimata dal classificatore f_j, che il campione $\mathbf{x}$ appartenga alla classe Cl_i, l'affidabilità del classificatore è data da:

$$\delta_j = max_i \; g_{ji}(\mathbf{x})$$

Gli outputs dei singoli classificatori h_i saranno raccolti con le relative affidabilità δ_i e l'output complessivo sarà dato dal primo di questi valori la cui metrica di affidabilità supera un valore di soglia prefissato.

2.3.5 Cascade generalization

Per **Cascade Generalization** si intende una tecnica di ensemble learning che consiste nell'utilizzo di più classificatori disposti sequenzialmente, estendendo ad ogni iterazione il dataset di partenza con nuovi attributi. Questa tecnica venne presentata da *João Gama* e *Pavel Brazdil* [15] nel 2000.

Un modello di Cascade Generalization parte prendendo in considerazione un dataset iniziale D costituito da coppie $(\mathbf{x}_n, y_n)$ e una sequenza di classificatori F_i. Dato in input al singolo classificatore F un campione $\mathbf{x}$, questo restituisce in output la distribuzione delle probabilità condizionali, ovvero un vettore $[p_1, ..., p_c]$, con c pari al numero di classi, dove

$$p_i = P(y = Cl_i|\mathbf{x})$$

rappresenta la probabilità che l'input $\mathbf{x}$ appartenga alla classe i. Si definisca l'operatore

$$\phi(\mathbf{x}, F)$$

come l'operatore che concatena al singolo campione $\mathbf{x}$ il vettore delle probabilità condizionate $[p_1, ..., p_c]$ restituito dal classificatore F, e l'operatore Φ come la versione dell'operatore ϕ applicata all'intero dataset.

Dato un dataset di addestramento L, un dataset di test T, e due classificatori F_1 e F_2, il classificatore F_1 del primo livello del modello di cascade, addestrato sul dataset L ($F_1(L)$), genera un nuovo dataset di addestramento e un nuovo dataset di test tramite l'applicazione dell'operatore Φ

$$Level_1train = \Phi(L, A(F_1(L), L))$$

$$Level_1test = \Phi(T, A(F_1(L), T))$$

dove l'operatore A restituisce il vettore delle probabilità condizionate. Il classificatore F_2 posto in sequenza al classificatore F_1, viene addestrato sul dataset di addestramento esteso $Level_1train$ e classifica i campioni del dataset di test esteso $Level_1test$

$$F_2 \nabla F_1 = A(F_2(\Phi(L, A(F_1(L), L))), \Phi(T, A(F_1(L), T)))$$

Una variante del Cascade Generalization prevede un'esecuzione in parallelo dei classificatori, e può essere rappresentata nel seguente modo:

$$\begin{aligned} F_n \nabla \{F_1, ..., F_{n-1}\} = A(F_n(\Phi(L, [A(F_1(L), L), ..., A(F_{n-1}(L), L)])), \\ (\Phi(T, [A(F_1(L), T), ..., A(F_{n-1}(L), T)]))) \end{aligned}$$

L'output complessivo sarà fornito dal classificatore F_n, addestrato secondo una delle due tecniche sopra illustrate, il quale riceverà in input le feature del dataset, concatenate alle probabilità condizionate ottenute dagli $n - 1$ classificatori precedenti.

2.3.6 Delegating

Per **Delegating** si intende una tecnica di ensemble learning presentata da *Cèsar Ferri*, *Peter A. Flach* e *Jose Hernandez-Orallo* [16].

Il *Cautious Classifier* (classificatore prudente) è l'entità che sta alla base del Delegating e consiste in un classificatore che classifica le istanze per le quali ha una confidenza sopra una certa soglia, delegando le restanti ad un altro classificatore. Ogni classificatore si ritroverà pertanto ad apprendere un numero di esempi inferiore rispetto al precedente classificatore e ogni esempio verrà classificato da un singolo classificatore. Il processo di delegazione termina quando non vi sono più istanze da delegare o quando è stato raggiunto un certo numero di steps di delegazione.

Dato un classificatore f e l'insieme delle classi C,

$$f_{CLASS}(e) = argmax_c f_{PROBc}(e)$$

rappresenta la classe assegnata all'esempio dal classificatore, mentre

$$f_{CONF}(e) = max_c f_{PROBc}(e)$$

rappresenta la confidenza della predizione.

Un aspetto importante consiste nel come viene determinata la soglia τ. Esistono diversi metodi che consentono di farlo, di particolare interesse sono il *Global Absolute Percentage* e lo *Stratified Absolute Percentage*.

- Nel *Global Absolute Percentage* la soglia τ è selezionata in modo tale che, scelta a priori una proporzione ρ dei dati di training, si massimizzi la confidenza di questi ultimi.
- Nello *Stratified Absolute Percentage* viene selezionata una soglia τ_c per ogni classe c.

L'output complessivo sarà fornito tramite votazione pesata dei singoli classificatori: ogni classificatore effettuerà una predizione alla quale sarà associato un peso, dovuto alla precisione del classificatore che l'ha computata; la predizione avente voto globale superiore a tutte le altre sarà fornita come output.

2.3.7 Arbitrating

Per **Arbitrating** si intende una tecnica di ensemble learning proposta da *Julio Ortega* [17]. Come nel Delegating, l'intuizione alla base dell'Arbitrating è che i classificatori utilizzati abbiano aree di competenza differenti, con la differenza che in questo caso i classificatori sono addestrati sull'intero dataset D e la specializzazione di ogni singolo classificatore viene valutata durante l'esecuzione da un modello detto *arbitro*.

L'*arbitro* è un modello di apprendimento, tipicamente un Decision Tree, che viene costruito in modo tale da apprendere l'area di competenza del classificatore corrispondente.

Dato un classificatore f e il training set ϵ formato dagli inputs e dalle corrispondenti classificazioni, l'algoritmo per computare l'arbitro corrispondente prevede i seguenti passaggi:

- il training set ε viene partizionato nel subset ε_C di esempi classificati correttamente dal classificatore f e nel subset ϵ_I di esempi classificati erroneamente;
- vengono selezionate le feature per costruire l'arbitro, in particolare queste feature devono includere almeno gli attributi primitivi che definiscono gli esempi e le classificazioni fatte dal classificatore;
- viene dunque costruito un albero decisionale T sulla base delle precedenti feature;
- in corrispondenza ad ogni foglia L dell'albero decisionale T, sulla base di C (numero di esempi in ε_C classificati dalla foglia) e I (numero di esempi in ϵ_I classificati dalla foglia), viene calcolata l'affidabilità come:

  $$\frac{max(C, I)}{C + I + \frac{1}{2}}$$

  Se $C > I$, la foglia L viene contrassegnata come "*corretta*", in caso contrario viene contrassegnata come "*errata*".

La predizione finale è quella fornita dal classificatore con maggiore affidabilità nell'area di input.

Chapter 3

Sintesi artificiale: l'apprendimento di una rete

> Scelte... forse Mr. Robot ha ragione; è di questo che si tratta, i si e i no della vita. Ma siamo noi a decidere? O decidono loro?
>
> *Elliot Alderson* - Mr. Robot

Lo sviluppo hardware dell'ultimo decennio ha reso le risorse di calcolo sempre più performanti e al contempo accessibili. Ciò ha fornito la possibilità di accedere agli strumenti per poter addestrare modelli di Machine Learning più semplicemente e ad un costo ridotto, consentendo così un'adozione più ampia del Machine Learning in molteplici ambiti, come l'Intrusion Detection. In precedenza sono state mostrate diverse tecniche di Machine Learning attraverso le quali è possibile raffinare il processo di classificazione fino ad ottenere modelli sempre più precisi e performanti.

In questo capitolo sarà presentata l'architettura di un Intrusion Detection System, basato sulle tecniche di Machine Learning presentate in precedenza, insieme ad una serie di algoritmi di Ensemble Learning allineati allo stato dell'arte, che implementano le tecniche esaminate nella Sez. 2.3, scelti sulla base della loro importanza, popolarità, performance e implementazione.

3.1 IDS: tra attacchi e classificazione

OMISSIS

3.2 Architettura dell'IDS

OMISSIS

3.3 Gli algoritmi: Ensemble e Gradient Boosting

OMISSIS

3.4 Clustering e visualizzazione dei dati

OMISSIS

Chapter 4

Componenti per un IDS

> [...] A quanto pare lei sta vivendo due vite distinte. In una di queste lei è Thomas A. Anderson programmatore per una rispettabile società informatica [...] l'altra vita lei la passa al computer, è una celebrità tra gli hacker con il soprannome di Neo, e di fatto ha commesso ogni crimine informatico concepibile e attualmente perseguito.
>
> *Agente Smith* - The Matrix

L'utilizzo di un dataset open-source è molto utile in fase di addestramento in quanto, grazie a questa scelta, è possibile valutare in maniera più agile le performance del modello implementato e promuovere la trasparenza e la riproducibilità dei risultati nell'ambito della cybersecurity.

In questo capitolo sarà preso in esame il **CSE-CIC-IDS2018**, ovvero il dataset scelto per la realizzazione dell'IDS.

4.1 Il Dataset: CSE-CIC-IDS2018

OMISSIS

4.2 Metriche per la valutazione del classificatore

OMISSIS

Chapter 5

Risultati sperimentali

> E se solo una macchina potesse sconfiggere un'altra macchina?
>
> *Alan Turing* - The Imitation Game

Nei capitoli precedenti sono state introdotte tutte le componenti necessarie alla realizzazione dell'IDS illustrato nel Cap. 3: in questo capitolo sarà preso in esame il dataset CIC-IDS2018 introdotto nella Sez. 4.1 e saranno mostrati i risultati dell'implementazione dell'IDS attraverso le tecniche di Ensemble Learning presentate nella Sez. 2.3, disponibili attraverso le implementazioni presentate nella Sez. 3.3.
OMISSIS

5.1 Preparazione dei dati

OMISSIS

5.2 Data cleaning

OMISSIS

5.3 Feature selection

OMISSIS

5.4 Clustering

OMISSIS

5.5 Scaling dei dati

OMISSIS

5.6 Addestramento

OMISSIS

5.7 Sistema conclusivo

OMISSIS

Chapter 6

Conclusioni

> La sicurezza informatica è un gioco tra gatto e topo: quando i criminali informatici sviluppano una nuova tecnica, gli esperti di sicurezza informatica devono sviluppare una nuova difesa.
>
> *Jeff Moss* - Hacker e fondatore di Black Hat e Def Con

Gli sviluppi nel campo del Machine Learning e gli avanzamenti hardware presentati durante il corso di questa trattazione, consentono di adattare rapidamente i sistemi di risposta alle minacce della rete allineandoli alle scoperte più recenti nel campo della Cybersecurity.

OMISSIS

Una carenza riscontrata nel settore durante lo studio dell'Intrusion Detection, è relativa alla presenza di dataset pubblicamente accessibili e con ampio contenuto informativo. I più recenti survey analizzati in materia di intrusion detection [45] [44] contrassegnano il CSE-CICIDS2018 presentato nel Cap. 4 come l'ultimo dataset pubblicamente disponibile per lo studio dell'argomento in esame. Tuttavia, si è riscontrato che oltre alla scarsità dei dataset in oggetto, quelli attualmente presenti ed open-source, sono spesso sbilanciati o non allineati allo stato dell'arte.
A tal proposito, un possibile miglioramento potrebbe coinvolgere l'utilizzo dei tool di raccolta dati con i quali i dataset disponibili sono stati generati (come ad esempio CICFlowMeter) al fine di generare delle estensioni dei dataset e rilasciarle pubblicamente, senza ledere però la privacy degli utenti delle reti coinvolte nella raccolta del traffico. Un'altra opportunità, sulla stessa scia, è la generazione di interi dataset basandosi sui medesimi strumenti di rilevazione, che colmino le lacune riscontrate attualmente e siano quindi bilanciati, inglobando anche attacchi più recenti e sofisticati.

OMISSIS

Complementarmente all'avanzamento nel campo dell'Intrusion Detection, un apporto significativo potrebbe essere dato dagli studi sul Meta-Learning. Il Meta-Learning potrebbe rappresentare in tal senso, la chiave di volta, consentendo l'addestramento di nuovi modelli di machine learning in grado di generalizzare sufficientemente, al punto da permettere l'apprendimento su più dataset contemporaneamente e fornire i mezzi per una discriminazione più generica di vaste famiglie di attacchi.

Appendix A

Decision Tree

Il *decision tree* è un algoritmo di apprendimento supervisionato usato sia per problemi di regressione che per problemi di classificazione.

Durante la fase di addestramento, l'algoritmo costruisce un albero dove ogni nodo interno, detto *nodo decisionale*, rappresenta una decisione da prendere rispetto ad un attributo dell'input, e ogni nodo foglia rappresenta la decisione finale. Di volta in volta, l'attributo scelto per fare lo split è quello con maggiore contenuto informativo ai fini di una migliore classificazione.

Una tecnica molto comune per la scelta dell'attributo sul quale effettuare lo split consiste nel valutare il *guadagno informativo*. Per spiegare in cosa consiste, è necessario introdurre il concetto di *Entropia*, definita come

$$I(P(v_1), ..., P(v_n)) = -\sum_{i=1}^{n} P(v_i) log_2 P(v_i)$$

dove n sono le singole classi e $P(v_i)$ rappresenta la probabilità che un campione appartenga alla classe i-esima.

Scelto un attributo A, il guadagno rappresenta la differenza tra l'entropia calcolata prima e dopo avere effettuato lo split su quell'attributo.
Si supponga che la variabile A possa assumere ν valori distinti, allora

$$remainder(A) = \sum_{i=1}^{\nu} P(\nu_i) I(P(\nu_i))$$

dove $P(\nu_i)$ rappresenta la probabilità che un campione assuma, in corrispondenza dell'attributo A, il valore i-esimo e $I(p(v_i))$ rappresenta l'entropia calcolata sui campioni che in corrispondenza dell'attributo A assumono valore i-esimo.

Sia il *guadagno informativo* (*IG(A)*) relativo all'attributo A definito come

$$IG(A) = I(D) - remainder(A)$$

l'attributo scelto per lo split è pertanto quello avente maggiore guadagno informativo.

Appendix B

SNE

La sigla SNE sta per *Stochastic Neighbor Embedding*, e consiste in un algoritmo di riduzione della dimensionalità ai fini di una migliore visualizzazione dei dati. Dati due punti x_i e x_j appartenenti ad uno spazio ad alta dimensionalità, la similarità tra questi due punti è data dalla probabilità condizionale $p_{j|i}$, la quale indica la probabilità che il campione x_j venga considerato vicino del campione x_i in base alla densità di probabilità di una distribuzione gaussiana in centrata in x_i.

$$p_{j|i} = \frac{\exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-||x_i - x_k||^2/2\sigma_i^2)} \quad \text{con } p_{i|i} = 0$$

Siano y_i e y_j le proiezioni nello spazio a bassa dimensionalità rispettivamente di x_i e x_j, la loro similarità è data dalla probabilità condizionale $q_{j|i}$ calcolata nel seguente modo:

$$q_{j|i} = \frac{\exp(-||y_i - y_j||^2)}{\sum_{k \neq i} \exp(-||y_i - y_k||^2)} \quad \text{con } q_{i|i} = 0$$

Lo scopo dell'algoritmo SNE è quello di trovare una proiezione dei dati in uno spazio a bassa dimensionalità che minimizzi la somma delle divergenze di Kullback-Leibler [46] tra le coppie di probabilità condizionali $p_{j|i}$ e $q_{j|i}$. La funzione costo è pertanto data dalla seguente formula:

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

La minimizzazione della funzione costo tramite il metodo di discesa lungo il gradiente, dove la componente del gradiente rispetto al punto y_i è data da

$$\frac{\delta C}{\delta y_i} = 2 \sum_{j \neq i} (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

Durante la fase di inizializzazione, le proiezioni nello spazio a bassa dimensionalità vengono ottenute campionando randomicamente dei punti da una gaussiana isotropica con bassa varianza centrata nell'origine.
Ad ogni iterazione il gradiente viene aggiornato secondo il metodo di discesa lungo il gradiente *con momento*

$$\Upsilon^{(t)} = \Upsilon^{(t-1)} + \eta \frac{\delta C}{\delta \Upsilon} + \alpha(t)(\Upsilon^{(t-1)} - \Upsilon^{(t-2)})$$

dove η rappresenta il learning rate e α l'iperparametro che controlla l'influenza del momento sull'aggiornamento del gradiente.

Bibliography

[1] M. Lottor. *Internet Growth (1981-1991)*. Tech. rep. Jan. 1992. DOI: 10.17487/rfc1296.

[2] S. Murugesan. "Understanding Web 2.0". In: *IT Professional* 9.4 (July 2007), pp. 34–41. DOI: 10.1109/mitp.2007.78.

[3] Y. Zhang, Y. Xiao, K. Ghaboosi, J. Zhang, and H. Deng. "A survey of cyber crimes". In: *Security and Communication Networks* 5.4 (July 2011), pp. 422–437. DOI: 10.1002/sec.331.

[4] "Anomaly-based network intrusion detection: Techniques, systems and challenges". In: 28.1-2 (Feb. 2009), pp. 18–28. DOI: 10.1016/j.cose.2008.08.003.

[5] D. Wolpert and W. Macready. "No free lunch theorems for optimization". In: *IEEE Transactions on Evolutionary Computation* 1.1 (Apr. 1997), pp. 67–82. DOI: 10.1109/4235.585893.

[6] Schmidhuber. "Learning to learn by gradient descent by gradient descent". In: *International Journal of Neural Systems* 1.2 (1987), pp. 141–148.

[7] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta. *Metalearning*. Springer Berlin Heidelberg, 2009. DOI: 10.1007/978-3-540-73263-1.

[8] T. Zoppi, M. Gharib, M. Atif, and A. Bondavalli. "Meta-Learning to Improve Unsupervised Intrusion Detection in Cyber-Physical Systems". In: *ACM Transactions on Cyber-Physical Systems* 5.4 (Sept. 2021), pp. 1–27. DOI: 10.1145/3467470.

[9] A. Shahraki, M. Abbasi, and Ø. Haugen. "Boosting algorithms for network intrusion detection: A comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost". In: *Engineering Applications of Artificial Intelligence* 94 (Sept. 2020), p. 103770. DOI: 10.1016/j.engappai.2020.103770.

[10] D. Gaikwad and R. C. Thool. "Intrusion Detection System Using Bagging Ensemble Method of Machine Learning". In: *2015 International Conference on Computing Communication Control and Automation*. IEEE, Feb. 2015. DOI: 10.1109/iccubea.2015.61.

[11] V. Sidharth and C. R. Kavitha. "Network Intrusion Detection System Using Stacking and Boosting Ensemble Methods". In: *2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)*. 2021, pp. 357–363. DOI: 10.1109/ICIRCA51532.2021.9545022.

[12] L. Breiman. "Bagging predictors". In: *Machine Learning* 24 (Aug. 1996), pp. 123–140. ISSN: 1573-0565. DOI: 10.1007/BF00058655.

[13] S. E. Robert. "The strength of weak learnability". In: *Machine Learning* 5 (June 1990), pp. 197–227. ISSN: 1573-0565. DOI: 10.1007/BF00116037.

[14] W. David. "Stacked Generalization". In: *Neural Networks* 5 (Dec. 1992), pp. 241–259. DOI: 10.1016/S0893-6080(05)80023-1.

[15] B. P. Gama João. "Cascade Generalization". In: *Machine Learning* 41 (Dec. 2000), pp. 315–343. DOI: 10.1023/A:1007652114878.

[16] H.-O. J. Ferri Cèsar Flach Peter. "Delegating Classifiers". In: Jan. 2004. DOI: 10.1145/1015330.1015395.

[17] J. Ortega, M. Koppel, and S. Argamon. "Arbitrating Among Competing Classifiers Using Learned Referees". In: *Knowledge and Information Systems* 3.4 (Nov. 2001), pp. 470–490. DOI: 10.1007/pl00011679.

[18] L. Breiman. In: *Machine Learning* 45.1 (2001), pp. 5–32. DOI: 10.1023/a:1010933404324.

[19] V. Agate, S. Drago, P. Ferraro, and G. Lo Re. "Anomaly Detection for Reoccurring Concept Drift in Smart Environments". In: *2022 18th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE. 2022, pp. 113–120.

[20] J. H. Friedman. "Greedy function approximation: A gradient boosting machine." In: *The Annals of Statistics* 29.5 (Oct. 2001). DOI: 10.1214/aos/1013203451.

[21] A. De Paola, P. Ferraro, S. Gaglio, G. Lo Re, and S. K. Das. "An adaptive bayesian system for context-aware data fusion in smart environments". In: *IEEE Transactions on Mobile Computing* 16.6 (2016), pp. 1502–1515.

[22] T. Chen and C. Guestrin. "XGBoost". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Aug. 2016. DOI: 10.1145/2939672.2939785.

[23] V. Agate, F. M. D'Anna, A. De Paola, P. Ferraro, G. Lo Re, and M. Morana. "A Behavior-Based Intrusion Detection System Using Ensemble Learning Techniques." In: *ITASEC*. 2022.

[24] A. De Paola, P. Ferraro, S. Gaglio, G. Lo Re, M. Morana, M. Ortolani, and D. Peri. "A context-aware system for ambient assisted living". In: *International Conference on Ubiquitous Computing and Ambient Intelligence*. Springer. 2017, pp. 426–438.

[25] V. Agate, F. Concone, A. De Paola, P. Ferraro, G. Lo Re, and M. Morana. "Bayesian Modeling for Differential Cryptanalysis of Block Ciphers: A DES Instance". In: *IEEE Access* 11 (2023), pp. 4809–4820.

[26] A. Timilsina, A. R. Khamesi, V. Agate, and S. Silvestri. "A Reinforcement Learning Approach for User Preference-aware Energy Sharing Systems". In: *IEEE Transactions on Green Communications and Networking* (2021).

[27] V. Agate, A. R. Khamesi, S. Silvestri, and S. Gaglio. "Enabling peer-to-peer User-Preference-Aware Energy Sharing Through Reinforcement Learning". In: *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*. 2020.

[28] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das. "The 1999 DARPA off-line intrusion detection evaluation". In: *Computer Networks* 34.4 (Oct. 2000), pp. 579–595. DOI: 10.1016/s1389-1286(00)00139-0.

[29] P. Ferraro and G. Lo Re. "Designing ontology-driven recommender systems for tourism". In: *Advances onto the Internet of Things*. Springer, 2014, pp. 339–352.

[30] V. Agate, A. De Paola, G. Lo Re, and M. Morana. "A platform for the evaluation of distributed reputation algorithms". In: *2018 IEEE/ACM 22nd International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*. IEEE. 2018, pp. 1–8.

[31] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani. "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization". In: *Proceedings of the 4th International Conference on Information Systems Security and Privacy*. SCITEPRESS - Science and Technology Publications, 2018. DOI: 10.5220/0006639801080116.

[32] V. Agate, A. De Paola, P. Ferraro, G. Lo Re, and M. Morana. "SecureBallot: A secure open source e-Voting system". In: *Journal of Network and Computer Applications* 191 (2021).

[33] I. Sharafaldin, A. Gharib, A. H. Lashkari, A. A. Ghorbani, and and. "Towards a Reliable Intrusion Detection Benchmark Dataset". In: *Software Networking* 2017.1 (2017), pp. 177–200. DOI: 10.13052/jsn2445-9739.2017.009.

[34] V. Agate, A. De Paola, G. Lo Re, and M. Morana. "Vulnerability Evaluation of Distributed Reputation Management Systems". In: *InfQ 2016 - New Frontiers in Quantitative Methods in Informatics*. ICST, Brussels, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016, pp. 1–8.

[35] A. De Paola, P. Ferraro, G. Lo Re, M. Morana, and M. Ortolani. "A fog-based hybrid intelligent system for energy saving in smart buildings". In: *Journal of Ambient Intelligence and Humanized Computing* 11.7 (2020), pp. 2793–2807.

[36] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. "SMOTE: Synthetic Minority Over-sampling Technique". In: (2011). DOI: 10.48550/ARXIV.1106.1813.

[37] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. "Lightgbm: A highly efficient gradient boosting decision tree". In: *Advances in neural information processing systems* 30 (2017).

[38] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin. *CatBoost: unbiased boosting with categorical features*. 2017. DOI: 10.48550/ARXIV.1706.09516.

[39] V. Agate, A. De Paola, G. Lo Re, and M. Morana. "A simulation software for the evaluation of vulnerabilities in reputation management systems". In: *ACM Transactions on Computer Systems (TOCS)* 37.1-4 (2021), pp. 1–30.

[40] A. Alsahaf, N. Petkov, V. Shenoy, and G. Azzopardi. "A framework for feature selection through boosting". In: *Expert Systems with Applications* 187 (Jan. 2022), p. 115895. DOI: 10.1016/j.eswa.2021.115895.

[41] L. van der Maaten and G. Hinton. "Viualizing data using t-SNE". In: *Journal of Machine Learning Research* 9 (Nov. 2008), pp. 2579–2605.

[42] V. Agate, F. Concone, and P. Ferraro. "A Resilient Smart Architecture for Road Surface Condition Monitoring". In: *The Proceedings of the International Conference on Smart City Applications*. Springer. 2021, pp. 199–209.

[43] P. G. Poličar, M. Stražar, and B. Zupan. "openTSNE: a modular Python library for t-SNE dimensionality reduction and embedding". In: *bioRxiv* (2019). DOI: 10.1101/731877. eprint: https://www.biorxiv.org/content/early/2019/08/13/731877.full.pdf.

[44] S. Walling and S. Lodh. "A Survey on Intrusion Detection Systems: Types, Datasets, Machine Learning methods for NIDS and Challenges". In: *2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. 2022, pp. 1–7. DOI: 10.1109/ICCCNT54827.2022.9984320.

[45] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho. "A survey of network-based intrusion detection data sets". In: *Computers and Security* 86 (Sept. 2019), pp. 147–167. DOI: 10.1016/j.cose.2019.06.005.

[46] S. Kullback and R. A. Leibler. "Information theory and statistics". In: *Journal of the American Statistical Association* 46.253 (1951), pp. 79–87.

[47] V. Agate, P. Ferraro, and S. Gaglio. "A Cognitive Architecture for Ambient Intelligence Systems". In: *AIC*. 2018, pp. 52–58.

[48] J. Wei and J. Wei. "Survey of network and computer attack taxonomy". In: *2012 IEEE Symposium on Robotics and Applications (ISRA)*. 2012, pp. 294–297. DOI: 10.1109/ISRA.2012.6219182.

[49] I. D. Mienye and Y. Sun. "A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects". In: *IEEE Access* 10 (2022), pp. 99129–99149. DOI: 10.1109/ACCESS.2022.3207287.