REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE ABDERAHMANE MIRA BEJAIA
FACULTE DES SCINCES EXACTES
DEPARTEMET INFORMATIQUE
OPTION : RESEAUX ET SECURITE
SYSTEME D'INFORMATION AVANCEE

Mémoire de Fin de Cycle

En vue de l'obtention du diplôme de Master Recherche
En Informatique
Option : Réseaux et Sécurité, Système d'Information Avancée

THEME

DETECTION D'INTRUSION A L'AIDE DES THECHNIQUES DE MACHINE LEARNING

Réalisées par :

M^lle^: ***LAIB SYLIA***

M^lle^: ***STAMBOULI SARA***

Soutenu devant le jury composé de :

Présidente	M^me^: **BATTAT**	Université de Béjaïa
Examinatrice	M^lle^: **F. CHERFIF**	Université de Béjaïa
Promotrice	M^me^: **M. YAICI**	Université de Béjaïa

Année universitaire : 2023/2024

Remerciement

On tient tout d'abord à remercier le bon Dieu le tout puissant et miséricordieux, qui nous a donné la force, la volonté et la patience d'accomplir ce modeste travail.

Nous exprimons nos profonds remerciements et tous nos respects pour nos très chers parents, qui nous ont éduqué, encadré et soutenu par tous les moyens, jours après jours, depuis notre naissance jusqu'à ce jour-là.

Nos sincères remerciements pour notre promotrice Mme

M. YAICI pour ses précieux conseils et ses aides durant toute la période du travail.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre travail en acceptant d'examiner notre mémoire et de l'enrichir par leurs recommandations.

On tient à remercier toutes les personnes qui ont contribué de près ou de loin au bon acheminement de ce travail.

Dédicace

Je dédie ce projet de fin d'études ....

Aux deux êtres les plus précieux à mon cœur, mes parents

"Yousef & Rezkia"

Aucun mot, aussi raffiné soit-il, ne saurait traduire l'immensité de l'amour que je vous porte ni la profondeur de la gratitude que je ressens pour tous les efforts et sacrifices que vous avez consentis inlassablement pour mon éducation et mon bien- être. Vos encouragements ont été le socle de ma quête de l'excellence académique, et vos critiques constructives ont façonné mon épanouissement personnel. J'espère avoir honorée les espoirs que vous avez nourris à mon égard. Ce modeste travail se veut un hommage sincère à votre soutien indéfectible, un témoignage de ma reconnaissance éternelle et de mon amour infini. Que le Tout- Puissant vous accorde santé, bonheur et longévité, afin que vous continuiez d'être le phare éclairant la voie de vos deux filles.

A ma seule et unique sœur, mon exemple "Farah"

Il m'est impossible de traduire par ces lignes l'étendue de mes sentiments d'amour et de tendresse à ton égard, ma chère grande sœur., J'espère avoir été à la hauteur de tes attentes. Je te souhaite une existence remplie de succès, de santé et de bonheur.

A ma grande mère "Djamila "

A ma chère amie et binôme"Sara "

A tous ceux qui m'ont dit non grâce aux je l'ai fait moi-même.

Sylia

Dédicace

Du fond de mon cœur, je dédie ce travail à ce qui me sont chers.
Je tiens à remercier mes parents et mes frères et sœur, ainsi que ma meilleure amie « Celia » qui ont été toujours une source d'inspiration et de soutien pour moi. Je tiens à vous dire que votre Amour inconditionnel et vos encouragements m'ont aidée à atteindre mes objectifs et à réaliser mes rêves.
Je voudrais également remercier ma chère amie et binôme « Sylia » pour sa contribution tout au long de ce travail. Nous avons travaillé dur pour réussir à mener ce mémoire à bien.
A tout personne qui m'aime, que j'aime.

Résumé

Ce mémoire explore l'utilisation de techniques de machine Learning pour améliorer la détection des intrusions dans les réseaux informatiques. L'objectif principal est de développer et d'évaluer des modèles prédictifs capables de distinguer les activités malveillantes des comportements normaux sur un réseau. La méthodologie comprend le prétraitement des données, la sélection des caractéristiques pertinentes, l'entraînement de différents algorithmes de machine Learning, et l'évaluation des performances des modèles. Les résultats montrent que les approches basées sur le machine Learning surpassent souvent les méthodes traditionnelles en termes de précision et de capacité à détecter des attaques variées. Ce travail met en lumière l'importance croissante de l'analyse prédictive dans la cyber sécurité, offrant des perspectives significatives pour renforcer la sécurité des réseaux contre les menaces modernes et émergentes.

Abstract

This thesis explores the use of machine learning techniques to improve intrusion detection in computer networks. The main objective is to develop and evaluate predictive models capable of distinguishing malicious activity from normal behaviour on a network. The methodology includes data pre-processing, selection of relevant features, training of different machine learning algorithms, and performance evaluation of the models. The results show that machine learning-based approaches often outperform traditional methods in terms of accuracy and ability to detect a variety of attacks. This work highlights the growing importance of predictive analysis in cybersecurity, offering significant prospects for strengthening network security against modern and emerging threats.

Table de matières

Chapitre I

Généralités sur la sécurité informatique et les attaques réseaux

Table de matières

Chapitre III

Etat de l'art

Chapitre IV

Contribution

Liste des figures

Liste des tableaux

Liste des Abréviation

IoT	Internet des Objets
IDS	Intrusion Detection System
IA	Intelligence Artificielle
NIDS	Network Intrusion Detection System
HIDS	Host Intrusion Detection System
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
ML	Machine Learning
PCA	Principal Component Analysis
DDoS	Distributed Denial of Service
SDN	Software Defined Network
LR	Logistic Regression
NB	Naive Bayes
DT	Decision Tree
RF	Random Forest
KNN	K-Nearest Neighbors
SVM	Support Vector Machine
RFE	Recursive Feature Elimination

Introduction générale

Introduction générale

Dans un monde de plus en plus connecté, la sécurité des réseaux informatiques est devenue une priorité absolue pour les entreprises et les institutions. Les systèmes de détection d'intrusions (IDS) jouent un rôle crucial dans la protection des infrastructures numériques en identifiant et en réagissant aux activités suspectes. Traditionnellement, ces systèmes reposaient sur des méthodes basées sur des signatures, mais ces approches montrent leurs limites face à l'évolution rapide des techniques d'attaque et à la sophistication croissante des cyberattaques.

L'avènement du machine Learning a ouvert de nouvelles perspectives pour la détection d'intrusions. Grâce à sa capacité à analyser de grandes quantités de données et à identifier des motifs complexes, le machine Learning offre des solutions prometteuses pour améliorer l'efficacité et la précision des IDS. En exploitant des algorithmes sophistiqués et des modèles d'apprentissage automatique, il est possible de détecter des anomalies et des comportements malveillants qui échappent aux méthodes traditionnelles.

Ce mémoire s'inscrit dans cette dynamique en explorant l'application des méthodes de machine Learning à la détection d'intrusions. Nous commencerons par une revue des techniques existantes, en mettant l'accent sur les approches basées sur le machine Learning. Ensuite, nous présenterons une analyse détaillée des ensembles de données utilisés pour l'évaluation des IDS, en particulier le NSL-KDD, qui est largement reconnu pour sa pertinence dans ce domaine. Nous décrirons également les principaux algorithmes de machine Learning utilisés pour la détection d'intrusions, tels que les forêts aléatoires, Arbre de décision, KNN et Régression Logistique.

Enfin, nous proposerons une série d'expérimentations visant à évaluer les performances de différentes approches de machine Learning appliquées à la détection d'intrusions. Les résultats obtenus seront analysés et comparés, afin de dégager les méthodes les plus efficaces et d'identifier les axes d'amélioration possibles.

En conclusion, ce mémoire vise à démontrer le potentiel des méthodes de machine Learning pour renforcer la sécurité des réseaux informatiques et à contribuer à l'évolution des systèmes de détection d'intrusions vers des solutions plus intelligentes et adaptatives.

Chapitre I

Généralités sur la sécurité informatique et les attaques réseaux

I.1. Introduction

La sécurité informatique représente désormais un enjeu crucial dans notre société interconnectée, où les réseaux informatiques jouent un rôle central dans notre quotidien, tant sur le plan personnel que professionnel.

Ce chapitre vise à fournir une compréhension approfondie des principes fondamentaux de la sécurité informatique ainsi que des types d'attaques les plus courantes qui menacent la confidentialité, l'intégrité et la disponibilité des systèmes informatiques et des données.

En explorant ces notions générales sur la sécurité informatique et les attaques réseau, nous serons mieux préparés à appréhender les défis auxquels font face les professionnels de la sécurité, et à mettre en place les mesures nécessaires pour protéger efficacement les infrastructures informatiques.

I.2. Définition de la sécurité informatique

La sécurité informatique englobe toutes les mesures prises pour minimiser la vulnérabilité d'un système face aux menaces, qu'elles soient accidentelles ou intentionnelles. Son objectif est de garantir **[1]** :

• **La confidentialité :** assurer que l'information ne soit accessible qu'aux personnes autorisées.

• **L'intégrité :** garantir que l'information ne puisse être altérée que par des individus autorisés.

• **La disponibilité :** veiller à ce que l'information soit accessible aux personnes autorisées.

I.3. Sécurité informatique, pour quoi faire

La décennie écoulée a été marquée par la migration implacable de toutes les facettes des opérations d'entreprise vers le domaine en ligne. Cette transition expose désormais chaque entreprise à un risque omniprésent de cyberattaque, visant à s'emparer d'informations cruciales telles que les données clients, les détails de paiement, les joyaux de la propriété intellectuelle, voire à ternir la réputation de l'entreprise.

De plus, l'essor généralisé du télétravail, la migration massive vers le cloud et la prolifération sans limites des appareils connectés offrent aux pirates informatiques et autres criminels du cyberespace des opportunités quasi illimitées d'attaques. Cette augmentation de la surface d'attaque, conjuguée à la sophistication croissante des adversaires numériques, contraint les entreprises à renforcer de manière incontournable et à actualiser constamment leurs pratiques de sécurité, avec un accent particulier sur la protection sans faille de leurs ressources basées dans le cloud.

Dans une certaine mesure, la sécurité informatique devient un impératif légal. En effet, dans certaines juridictions, la loi oblige les entreprises à investir de manière substantielle dans le développement et la mise en œuvre de concepts de sécurité informatique **[1]**.

I.4. Type des sécurités informatiques

La sécurité informatique englobe de manière générale tout ensemble de plans, mesures ou outils visant à sauvegarder les ressources numériques d'une entreprise **[1]**. Cette discipline comprend divers éléments :

I.4.1. Cyber sécurité

Vise à garantir la protection des ressources numériques, englobant les réseaux, les systèmes, les ordinateurs, les données, etc., en les préservant contre les cyberattaques **[1]**.

I.4.2. Sécurité des points d'extrémité

Adopte une approche axée sur la défense des end points (points d'extrémité) tels que les ordinateurs de bureau, les ordinateurs portables et les terminaux mobiles contre les activités malveillantes **[1]**.

I.4.3. Sécurité du cloud

Englobe la stratégie et les solutions visant à protéger l'infrastructure cloud contre les cyberattaques, couvrant également tous les services ou applications hébergés dans l'environnement cloud **[1]**.

I.4.4. Sécurité des applications

Englobe la stratégie et les solutions visant à protéger l'infrastructure cloud contre les cyberattaques, couvrant également tous les services ou applications hébergés dans l'environnement cloud **[1]**.

I.4.5. Sécurité du réseau

Englobe l'utilisation d'outils, de technologies et de processus pour protéger le réseau et l'infrastructure critique contre les cyberattaques et les activités malveillantes **[1]**.

I.4.6. Sécurité des conteneurs

Représente le processus continu de protection des conteneurs, y compris le pipeline des conteneurs, l'infrastructure de déploiement et la Supply Chain, contre les cyberattaques **[1]**.

I.4.7. Sécurité de l'IoT

Branche spécialisée de la cyber sécurité qui se consacre à la protection, à la surveillance et à la neutralisation des menaces visant l'Internet des objets (IoT) et le réseau de terminaux IoT connectés qui collectent, stockent et partagent des données via Internet **[1]**.

I.5. Politique de sécurité informatique

Pour atteindre les objectifs de sécurité énoncés, il est impératif de mettre en place une politique de sécurité robuste qui englobe l'ensemble des règles et directives visant à protéger les ressources et les informations contre tout préjudice pouvant affecter leur confidentialité, intégrité et disponibilité.

Elle se matérialise sous forme de lois et de consignes qui définissent clairement les sujets, les objets, ainsi que les activités autorisées et interdites **[2]**.

I.5.1. Étapes types dans l'établissement d'une politique de sécurité [2]

• **Première étape :** Identification des vulnérabilités ;

• **Deuxième étape :** Évaluation des probabilités associées à chacune des menaces ;

• **Troisième étape :** Évaluation du coût d'une intrusion réussie ;

• **Quatrième étape :** Choix des contre-mesures ;

• **Cinquième étape :** Évaluation des coûts des contre mesure ;

• **Sixième étape :** Décision.

I.5.2. Fonctions de sécurité

La politique de sécurité utilise un catalogue de fonctions de sécurité, parmi lesquelles on peut trouver **[2]** :

• **L'identification des sujets :** l'identification des sujets, des objets et des opérations effectuées par ces sujets sur ces objets.

• **L'authentification :** est la preuve de l'identité de ces entités ou de ces opérations.

• **L'intimité numérique :** est une fonction qui consiste à abriter l'identité d'une entité et ses activités, en masquant son observation.

• **La traçabilité :** est une fonction qui consiste à repérer l'histoire des entités.

• **L'audit du système :** l'observation, l'enregistrement, l'analyse et la compréhension des événements importants ou anormaux qui vont concourir à reconstituer le fil de son histoire, après une panne ou d'une attaque.

• **L'imputabilité :** est une fonction qui permet de garantir qu'une communication ou une transaction ne peut être niée, ni à l'émission, ni à la destination par ses responsables

• **L'autorisation des actions :** l'autorisation des actions par un sujet sur des objets. Les droits spécifiques et les privilèges des sujets sont définis par cette fonction.

• **Le contrôle d'accès :** est la restriction d'accès aux ressources et aux informations, aux seuls sujets qui sont autorisés.

• **La protection des contenus :** est de cacher la signification des informations aux sujets non autorisés, en utilisant des primitives cryptographiques.

• **La gestion de sécurité :** est la gestion du cycle de vie de toutes les fonctions précédentes, essentiellement la configuration et la protection de ces fonctions de sécurité.

I.6. Contre qui faut-il se protéger

Aujourd'hui, la nécessité de protection s'impose face à trois éléments **[2]** :

I.6.1. Menaces

Une violation potentielle de la sécurité, c'est-à-dire un signe qui laisse prévoir un danger. On trouve deux types de menaces **[2]** :

• **Menaces accidentelles :** bugs logiciels, les pannes matérielles et autres défaillances incontrôlables ;

• **Menaces intentionnelles :** Menace Active : Menace de modification non autorisée et Menace Passive : Menace de divulgation non autorisée.

I.6.2. Attaquants

Il est possible de classifier en deux catégories **[2]** :

• **Hackers :** L'hacker n'a pas de motivations réelles, on peut qualifier ses actions de "passe-temps" ;

• **Crackers :** Le cracker a des motivations réellement criminelles, pour but d'accéder illégalement à des systèmes, des réseaux, des logiciels ou des données.

I.6.3. Attaques

Une attaque est une concrétisation d'une menace, Nous allons à présent débuter la suite de notre chapitre en abordant les détails sur les Attaque réseaux.

I.7. Définition des attaques réseaux

Les attaques réseaux sont des actions malveillantes menées au moyen d'un réseau informatique visant à causer un dommage aux informations et aux personnes qui les traitent. Elles s'appuient sur des vulnérabilités liées directement aux protocoles ou à leur implémentation. Dans **[3]**, Ghorbani et al définissent les attaques réseaux comme l'activité malicieuse visant l'interruption, la dégradation, la perturbation de services accessibles aux travers d'un réseau. L'objectif de ces attaques est de porter atteinte à l'intégrité, la disponibilité ou la confidentialité de ces services. Les attaques sont diverses et variées, et peuvent cibler une machine ou un système complet. Quant à d'autre savons, ils définissent les attaques réseaux comme des attaques visant un réseau ou des utilisateurs en manipulant les protocoles réseaux de la couche physique à la couche application.

I.8. Historique des grandes attaques les plus connues

• En décembre 1996, Ping Of Death est une attaque historique de réseau, elle entraîne un arrêt immédiat des systèmes vulnérables **[4]** ;

• En octobre 1997, Land est une attaque réseau, utilisant l'usurpation d'adresse IP afin d'exploiter une faille de certaines implémentations du protocole TCP/IP dans les systèmes **[5]**;

• En juin 1998, l'attaque Syndrop basé sur Teardrop en TCP avec le bit SYN et des champs invalides tel que le numéro de séquence, la taille de fenêtre ;

• En mars 1999, Melissa est un virus infectant qui attaque les messageries Outlook via des pièces jointes contaminées **[6]** ;

• En octobre 2002, Ping Flood est une cyberattaque visant différents systèmes connectés à Internet ;

• En juin 2010, le monde découvrait l'existence de Stuxnet un malware ayant pour objectif de saboter le programme nucléaire iranien **[7]** ;

• En août 2012, une variante du malware Shamoon a infecté le système d'information du géant de l'industrie pétrolière italienne Saipem**[8]** ;

• En octobre 2013, Yahoo a subi une énorme violation de données **[9]** ;

• En novembre 2014, la société américaine Sony Pictures Entertainment, filiale du groupe japonais Sony, est victime d'un piratage massif de ses données et la révélation des informations privées telle que les salaires de cadres supérieurs **[10]** ;

• En septembre 2016, les auteurs du logiciel malveillant Mirai ont lancé une attaque DDoS sur le site web d'un expert en sécurité bien connu **[11]** ;

• En mai 2017, la première attaque WannaCry qu'est une faille du système Microsoft Windows qui avait été exploitée ;

• En juin 2017, une nouvelle vague massive de cyberattaques mondiales rappelant le mode d'action du virus WannaCry survenu le week-end du 12 au 13 mai 2017 ;

• En mai 2020, la compagnie aérienne LowCostEasy Jet easyJet a été victime d'une attaque de hackers (très sophistiquée), qui leur a permis d'accéder aux données personnelles d'environ 9 millions de clients **[12]** ;

• En août 2021, c'est le centre hospitalier d'Arles qui a été touché par une attaque cybercriminelle (ransomware) ciblant son système d'information qui ralentit le fonctionnement des services **[13]**.

I.9 Type des attaques informatiques

On distingue trois types d'attaque informatique connue aujourd'hui **[14]** :

I.9.1. Attaques direct

C'est la plus simple des attaques. L'hacker attaque directement sa victime à partir de son ordinateur. En effet, les programmes de hack qu'ils utilisent ne sont que faiblement paramétrables, et un grand nombre de ces logiciels envoient directement les paquets à la victime.

I.9.2. Attaques par rebond

Consistant à attaquer une machine par l'intermédiaire d'une autre machine, afin de masquer les traces permettant de remonter à lui, dans le but d'utiliser les ressources de la machine servant de rebond.

I.9.3. Attaques indirectes par réponse

Cette attaque est un dérivé de l'attaque par rebond. Elle offre les mêmes avantages, mais au lieu d'envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, l'attaquant va lui envoyer une requête. Et c'est cette réponse à la requête qui va être envoyée à l'ordinateur victime.

I.10. Différentes classes d'attaques informatiques

On distingue trois classes d'attaque informatique :

I.10.1. Classification selon l'effet de l'attaque

Selon les effets résultant de l'attaque on peut classifier les attaques en deux groupes principaux : les attaques passives et les attaques actives.

• **Les attaques passives :** consistent à accéder, utiliser ou à observer le système cible sans modifier les données ou dysfonctionner les ressources de ce dernier, elles sont généralement indétectables.

• **Les attaques actives :** consistent à effectuer des changements non autorisés sur les données des systèmes, à s'introduire dans des équipements réseau ou à perturber leurs fonctionnements, les attaques de ce type sont bien évidemment plus dangereuses.

I.10.2. Classification selon la source de l'attaque

En termes de relation intrusion-victime, les attaques sont classées comme suit :

• **Les attaques internes :** provenant des employés de leur entreprise ou de leurs partenaires commerciaux ou clients,

• **Les attaques externes :** venant de l'extérieur, fréquemment via Internet.

I.10.3. Classification selon la cible de l'attaque

• **Les attaques réseaux :** Les attaques réseau exploitent les failles liées directement aux protocoles ou à leur mise en œuvre. Bien qu'il en existe un grand nombre, la plupart sont des variantes des cinq attaques réseau (définie ci-dessus de 1 à 5) les plus couramment connues aujourd'hui **[15]** ;

• **Les attaques applicatives :** Les attaques applicatives s'appuient principalement sur des vulnérabilités spécifiques aux applications utilisées **[15]**.

I.11. Attaques réseaux

On distingue plusieurs types d'attaque réseaux connues aujourd'hui :

I.11.1. Attaques par fragmentation (Fragments attacks)

Les attaques par fragmentation IP sont une forme courante d'attaque par déni de service, dans laquelle l'auteur domine un réseau en exploitant les mécanismes de fragmentation des datagrammes. Cette attaque outrepasse la protection des équipements de filtrage IP. Ces attaques étant historiques, les pare-feu actuels les prennent en compte depuis longtemps dans leur implémentation **[15]**.

I.11.2. Usurpation d'adresse IP (IP Spoofing)

L'usurpation d'adresse IP, fait référence à la création de paquets IP avec une fausse adresse IP source pour se faire passer pour un autre système informatique **[15]**.

Certaines des attaques populaires lancées par l'usurpation d'adresse IP sont :

• **Usurpation d'identité aveugle :** implique qu'un pirate extérieur envoie des paquets à sa cible pour obtenir des numéros de séquence. Sans connaissance du réseau cible, le pirate force la machine à répondre pour analyser ces numéros. Avec ces informations, le pirate peut falsifier son identité en injectant des données dans le flux de paquets sans authentification.

• **Usurpation d'identité non aveugle :** Lorsque le pirate réside sur le même sous-réseau que sa cible, il peut écouter les transmissions existantes pour comprendre un cycle de séquence/d'accusé de réception entre sa cible et un autre hôte, ce qui signifie qu'il n'est pas "aveugle". Une fois qu'il connaît le numéro de séquence, le pirate peut détourner des sessions déjà établies en se faisant passer pour une autre machine, contournant ainsi toute authentification précédemment effectuée sur cette connexion.

I.11.3. Capture de session TCP (TCP Session Hijacking)

Le détournement de session est un moyen d'obtenir un contrôle total ou partiel sur une connexion TCP/IP établie. Cette attaque repose sur une connexion de confiance entre deux parties communicantes pour être en place, puis fonctionne soit pour modifier le paquet circulant entre les machines, soit pour prendre la place de l'un des deux parties **[15]**.

I.11.4. Usurpation ARP (ARP Spoofing)

L'usurpation d'identité ARP se produit sur un réseau local à l'aide d'un ARP. Un ARP est un protocole de communication connectant une adresse de protocole Internet dynamique à une adresse de machine physique **[15]**.

I.11.5. Usurpation DNS (DNS Spoofing)

L'usurpation de serveur de noms de domaine est une attaque dans laquelle des enregistrements DNS modifiés sont utilisés pour rediriger le trafic en ligne vers un site Web frauduleux qui ressemble à la destination prévue.

Une fois sur place, les utilisateurs sont invités à se connecter à leur compte, ce qui donne à l'auteur la possibilité de voler leurs identifiants d'accès et d'autres types d'informations sensibles **[15]**.

I.11.6. Capture de données (Sniffing)

Le renfilage de paquets est l'interception de paquets de données traversant un réseau. Un programme snifer fonctionne au niveau de la couche Ethernet en combinaison avec la carte d'interface réseau pour capturer tout le trafic circulant vers et depuis le site hôte Internet.

I.11.7. Ecoute clandestine (Eavesdroppingattack)

Elle se produit lorsqu'un attaquant surveille ou écoute le trafic réseau en transit, puis interprète toutes les données non protégées. Avant d'attaquer un réseau, les attaquants souhaitent connaître l'adresse IP des machines du réseau, les systèmes d'exploitation qu'elles utilisent et les services qu'elles proposent.

I.11.8. Déni de service (DoS) déni de service distribué (DDoS)

DoS et DDoS est un type d'attaque très courant. L'attaquant empêche un serveur de fournir des services, entraîne un retard infini rend et les ressources en ligne indisponibles pour les utilisateurs prévus **[16]**.

I.11.9. Attaque Smurf (Smurf Attack)

Attaque Smurf est une forme d'attaque par déni de service distribué qui rend les réseaux informatiques inopérants. L'attaquant crée de nombreux paquets ICMP avec l'adresse IP de la victime visée comme IP source et diffuse ces paquets dans un réseau informatique à l'aide d'une adresse de diffusion IP **[17]**.

I.11.10. Inondation SYN (SYN Flood)

SYN Flood se produit lorsqu'un client tente de démarrer une connexion TCP avec un serveur, le client et le serveur échangent une série de messages qui s'exécutent. Une attaque DDoS par inondation SYN exploite une faiblesse connue dans une séquence de connexion TCP dans laquelle une demande SYN pour initier une connexion TCP avec un hôte doit être répondue par une réponse SYN-ASK de cet hôte, puis confirmée par une réponse ACK du demandeur **[19]**.

I.12. Conclusion

En conclusion, la sécurité informatique et la lutte contre les attaques réseau sont des préoccupations cruciales pour toute organisation ou individu opérant dans un environnement numérique. Ce chapitre a abordé certains des principes fondamentaux de la sécurité informatique ainsi que des types d'attaques réseau courants. Cependant, ce n'est qu'un premier pas dans la compréhension de ces sujets complexes. Pour aller plus loin, il est essentiel d'explorer en détail les techniques de protection, les meilleures pratiques de sécurité et les dernières tendances en matière de cyber sécurité. Le prochain chapitre abordera ces aspects en fournissant des connaissances approfondies et des conseils pratiques pour renforcer la sécurité des systèmes informatiques et prévenir les attaques réseau.

Chapitre II

Méthodes de l'intelligence artificielle pour la détection et la prévention des attaques réseaux

II.1. Introduction

IDS basés sur l'IA se révèlent être des outils puissants dans la lutte contre les attaques informatiques. En combinant les capacités de détection automatisée de l'IA avec une analyse intelligente des comportements anormaux, ces systèmes sont capables de détecter et de répondre efficacement aux menaces en temps réel. Dans ce chapitre, nous explorerons le fonctionnement, les avantages et les défis des IDS basés sur l'IA, ainsi que leur rôle essentiel dans la protection des infrastructures numériques contre les attaques malveillantes.

II.2. Définition des Systèmes de Détection d'Intrusion

Les Systèmes de Détection d'Intrusion (IDS pour Intrusion Detection System) constituent des éléments cruciaux de la sécurité des réseaux, s'intégrant au sein des infrastructures informatiques. Leur rôle consiste à surveiller l'activité des systèmes ou des réseaux afin de repérer toute intrusion potentielle et d'émettre des alertes en cas d'activités suspectes. Les IDS peuvent surveiller tout ou partie des réseaux, visant à détecter les attaques avec un haut niveau de précision tout en minimisant les fausses alarmes **[18]**.

II.3. Fonctionnement d'IDS

Un IDS est basé sur trois aspects fonctionnels, à savoir **[18]** :

• **Moteur d'analyse :** qui trouve des signes d'intrusion ;

• **Source d'information :** qui fournit un flux d'enregistrements d'événements y compris les pistes d'audit, le trafic réseau, la trace des appels système, etc ;

• **Composant de réponse :** qui génère des réactions basées sur les résultats du moteur d'analyse.

La **Figure 1** représente le processus de détection d'intrusion. Pour identifier une intrusion, un IDS effectue les tâches suivantes :

1) collecte des données ;

2) prétraitement des données ;

3) reconnaissance de l'intrusion ;

4) mise en œuvre de mesures correctives.

Les données sont collectées à partir d'une ou plusieurs sources de données. Un prétraitement est effectué sur les données collectées et seront transférées dans un format compréhensible par le composant de détection. Ce dernier est utilisé pour caractériser le comportement intrusif en utilisant plusieurs techniques et algorithmes. Enfin, le composant de réponse signale l'intrusion et éventuellement les informations temporelles correspondantes.

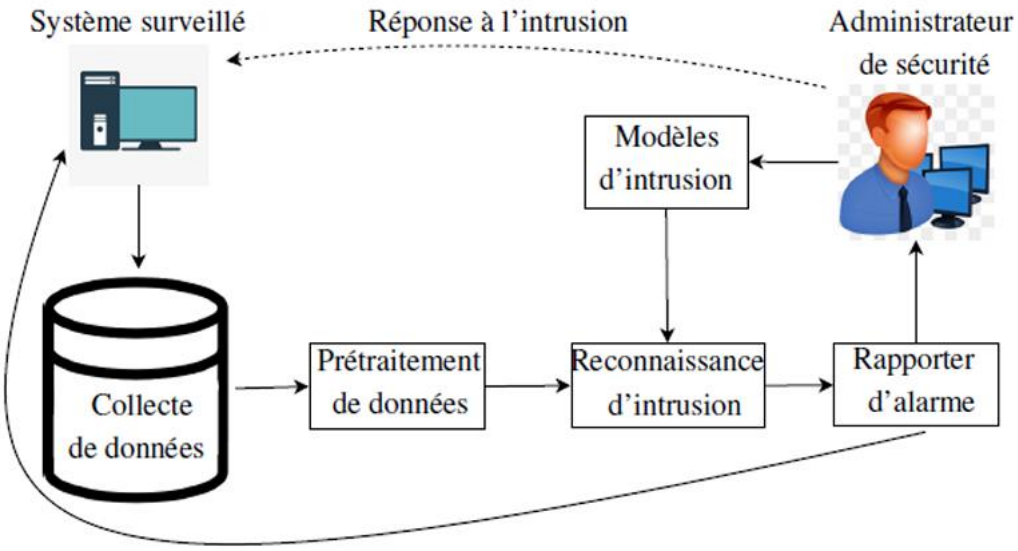

Figure II.1 : Processus de détection d'intrusion **[18]**.

II.4. Types d'IDS

Les IDS peuvent se classer selon trois catégories majeures **[18]** :

II.4.1. IDS réseaux (Network-based IDS (NIDS))

Les IDS réseau analysent et interprètent les paquets qui circulent sur un réseau ou un segment spécifique afin d'identifier les paquets potentiellement malveillants. Ils analysent minutieusement chaque trame à travers ses différentes couches, y compris les couches réseau, transport et application. Grâce à la décomposition des paquets et à leur connaissance des protocoles, les NIDS sont capables de repérer les paquets suspects conçus pour contourner les pare-feu.

II.4.2. IDS hôtes (Host-based IDS (HIDS))

Les systèmes de détection d'intrusion basés sur l'hôte se concentrent spécifiquement sur les activités liées à l'hôte sur lequel ils sont installés, dans le but de détecter toute activité suspecte. Ils exploitent diverses sources de données telles que les journaux d'audit de sécurité, les journaux système et le trafic réseau de l'hôte pour identifier les comportements anormaux. Les HIDS sont généralement déployés sur des hôtes critiques tels que les serveurs contenant des données hautement sensibles et les serveurs accessibles au public.

II.4.3. IDS hybrides (Hybrid-Based IDS)

Une évolution récente dans le domaine de la détection d'intrusion consiste à fusionner les NIDS et les HIDS afin de créer des IDS hybrides. Ces systèmes de détection d'intrusion hybrides offrent une flexibilité accrue et renforcent la sécurité globale. En combinant les capacités de détection des systèmes IDS déployés à différents niveaux du réseau, ils sont en mesure de détecter efficacement les attaques ciblant des éléments spécifiques ainsi que celles visant l'intégralité du système.

II.5. Critères d'évaluation d'IDS

Voici les critères les plus importants utilisés pour une évaluation plus réaliste des systèmes de détection d'intrusion **[18]** :

II.5.1. Erreurs de classification

Voici les critères les plus importants utilisés pour une évaluation plus réaliste des systèmes de détection d'intrusion :

• **Vrais positifs (True Positive (TP)) :** se produisent lorsqu'un IDS classifie correctement une intrusion ;

• **Vrais négatifs (True Negative (TN)) :** se produisent lorsqu'un événement normal est correctement classé comme une action légitime ;

• **Faux positifs (False Positive (FP)) :** se produisent lorsque le système reconnaît à tort que des actions légitimes sont une intrusion ;

• **Faux négatifs (False Negative (FN)) :** se produisent lorsqu'un IDS classifie à tort des intrusions comme des actions légitimes.

		Résultat de prédictions	
		Légitime	Intrusion
Evènement actuel	Légitime	**TN**	**FP**
	Intrusion	**FN**	**TP**

Tableau II.1 : Matrice de confusion.

II.5.2. Exactitude (Accuracy)

C'est une déclaration de l'exactitude du fonctionnement d'un IDS, mesurant le pourcentage de détection et d'échec ainsi que le nombre de fausses alarmes que le système produit.

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+FP+TN+FN}} * 100 \qquad \textbf{(II.1)}$$

II.5.3. Précision (Precision)

Il s'agit d'une métrique indiquant combien d'événements, qui sont prédits par un IDS comme étant intrusifs, sont des intrusions réelles.

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} * 100 \qquad \textbf{(II.2)}$$

II.5.4. Rappel (Recall)

Il mesure la partie manquante de la précision, c'est-à-dire là partir de toutes les événements normaux, combien sont correctement classées par le système.

$$\boldsymbol{Recall} = \frac{TP}{TP+FN} * 100 \qquad \textbf{(II.3)}$$

II.5.5. F1 score

Étant donné que la précision et le rappel ne définissent pas complètement l'exactitude d'un IDS, alors il est plus approprié d'utiliser une combinaison de celles-ci.

$$\boldsymbol{F1-score} = \frac{Precision*Recall}{Precision+Recall} * 2 \qquad \textbf{(II.4)}$$

II.6. Avantages et limites d'IDS

II.6.1. Avantages d'IDS

- Rapporter les intrusions et les tentatives d'intrusion survenues ;

- Détecter les attaques et les violations de sécurité qui échappent à d'autres systèmes de sécurité ;

- Prendre des mesures proactives telles que le blocage des adresses IP suspectes ou l'interruption de connexions compromettantes ;

- Fournir des lignes directrices essentielles pour élaborer la politique de sécurité de l'organisation.

II.6.2. Limites d'IDS

- Il est limité face à des volumes élevés et des vitesses de trafic Internet élevées ;

- Il ne peut pas garantir une protection totale contre tous les types d'attaques, certains types d'attaques peuvent passer inaperçus et ils peuvent même être la cible d'attaques ;

- Son fonctionnement n'est pas entièrement automatisé, nécessitant une importante allocation de ressources humaines pour la gestion ;

- Les IDS ont tendance à générer un nombre excessif de faux positifs.

Dans ce qui suit nous allant travailler sur les NIDS et on va abordes les méthodes Machine Learning utilisés pour les NIDS.

II.7. Intelligence Artificielle

• **Intelligence artificielle (IA) :** comprend l'ensemble des techniques et des théories visant à créer des modèles capables de reproduire le comportement humain. Ce domaine englobe diverses technologies, telles que l'apprentissage automatique (Machine Learning) et l'apprentissage profond (Deep Learning) **[19]** ;

• **Machine Learning (ML) :** est une branche de l'IA qui englobe toutes les méthodes et algorithmes permettant aux machines d'apprendre de manière automatique à partir de modèles mathématiques. Cette discipline implique l'utilisation de données passées, telles que des historiques, pour créer des algorithmes de prédiction en vue de données futures **[19]** ;

• **Deep Learning (DL) :** est une subdivision du Machine Learning basé sur des réseaux de neurones qui permettent à une machine de s'entrainer elle-même pour effectuer une tâche **[19]**.

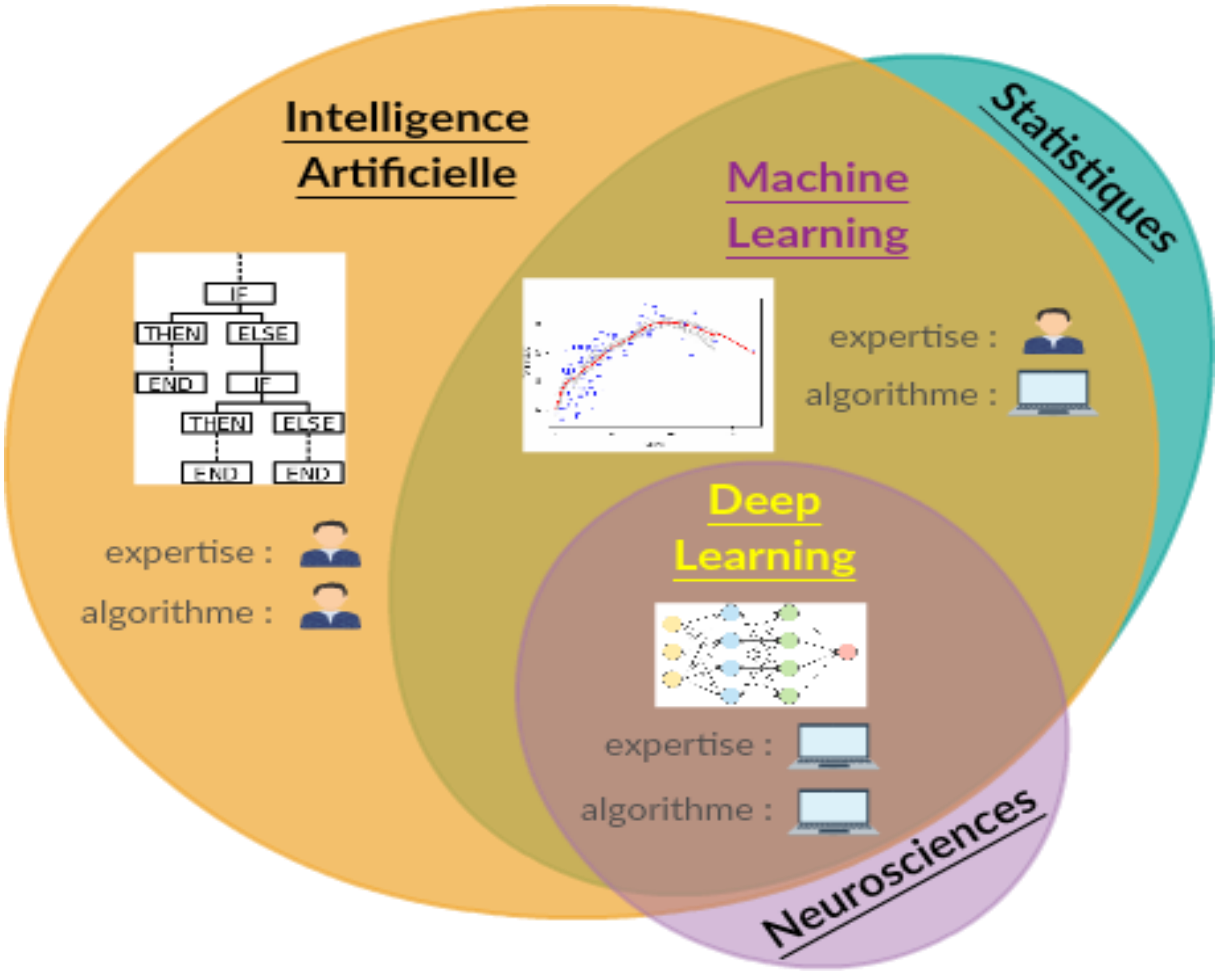

Figure II.2 : les liens et interactions entre ces 3 disciplines **[20]**.

II.8. Techniques de Machine Learning

Pour doter un ordinateur de la capacité d'apprendre, on a recouru à des méthodes d'apprentissage largement inspirées de la façon dont les êtres humains acquièrent de nouvelles compétences. Le processus d'apprentissage automatique peut être supervisé ou non supervisé **[21]**.

II.8.1. Apprentissage automatique supervisé (Supervised Learning)

L'apprentissage automatique peut être supervisé dans le cas où des étiquettes sont associées à des données d'apprentissage, et vous essayez de prédire des étiquettes pour des données futures. Avec le Supervised Learning on peut développer des modèles pour résoudre 2 types de problèmes **[21]** :

• **Les problèmes de Régression :** on cherche à prédire la valeur d'une variable continue, c'est-à-dire une variable qui peut prendre une infinité de valeurs (prédire le prix d'un appartement (y) selon sa surface habitable (x)).

• **Les problèmes de Classification :** on cherche à classer un objet dans différentes classes, c'est-à-dire que l'on cherche à prédire la valeur d'une variable discrète qui ne prend qu'un nombre fini de valeurs (Prédire si un émail est un spam (classe y = 1) ou non (classe y = 0) selon le nombre de liens présent dans l'émail (x)).

II.8.2. Apprentissage automatique non supervisé (Unsupervised Learning)

L'apprentissage automatique peut être non supervisé dans le cas où des étiquettes ne sont pas associées à des données d'historique. On peut ne même pas savoir ce que sont les étiquettes que vous essayez de prédire. Le clustering, la détection d'anomalie, etc. sont une forme typique d'apprentissage non supervisé **[21]**.

II.8.3. Apprentissage automatique semi-supervisé (Semi-supervised learning)

L'apprentissage semi-supervisé est une technique d'apprentissage automatique qui consiste à partir d'un jeu de données constitué en majorité de données non labellisées et en minorité de données labellisées. Il se situe entre l'apprentissage supervisé qui utilise des données labellisées et l'apprentissage non supervisé qui utilise des données non labellisées **[21]**.

II.9. Type d'algorithme supervisé

Dans l'apprentissage supervisé, on a deux types d'algorithmes **[21]** :

• **Les algorithmes de régression :** qui cherchent à prédire une valeur continue, une quantité.

• **Les algorithmes de classification :** qui cherchent à prédire une classe/catégorie.

II.9.1. Arbre de décision (DecisionTree (DT))

II.9.1.1. Définition

Arbre de Décision (DT), est l'un des algorithmes Machine Learning supervisé de base utilisé pour classification et régression d'un ensemble de données donné en appliquant une série de décisions (règles). Ce modèle adopte une structure arborescente composée de nœuds, de branches et de feuilles. Chaque nœud représente un attribut (une caractéristique), chaque branche représente une décision (une règle), et chaque feuille correspond à un résultat possible (une étiquette de classe) **[22]**.

II.9.1.2. Modèle d'arbre de décision

Pour bien comprendre la stratégie adoptée par les arbres de décision, il y a 4 étapes typiques :

• La première étape consiste à subdiviser l'ensemble de données original en deux sous-ensembles, suite à la première subdivision, nous aurons comme résultat deux sous-ensembles de données ;

• Les sous-ensembles seront à nouveau subdivisés sur la base d'autres conditions, à chaque étape, la condition qui fournit la meilleure bipartition du sous-ensemble original est choisie ;

• La division se déroule de manière itérative. Il est donc nécessaire de définir une condition d'arrêt ;

• A chaque itération, l'algorithme génère une structure arborescente dont les nœuds représentent les choix faits à chaque fois, chaque feuille contribuant à la classification globale des données d'entrée.

II.9.2. Machine à vecteur de support (Support Vector Machine (SVM))

II.9.2.1. Définition

Machine à vecteurs de support (SVM), est une technique de classification supervisée qui opère en identifiant un hyperplan de séparation dans un espace de caractéristiques. Elle est employée pour résoudre des problèmes à la fois linéaires et non linéaires.

L'objectif est de déterminer un hyperplan qui sépare de manière optimale les deux classes, afin de maximiser la marge entre celles-ci **[23]**.

Ps : l'hyperplan est défini par une équation linéaire : **ax + b = 0**, **a :** vecteur de poids (définit la direction de l'hyperplan), **b :** biais (détermine la position de l'hyperplan dans l'espace)).

II.9.2.2. Processus d'entraînement d'un SVM

• **Transformation des données :** convertir les données a des vecteurs de caractéristiques numériques ;

• **Entraînement du modèle :** Entraîner le SVM en utilisant l'ensemble de données d'entraînement. Le modèle trouve l'hyperplan qui maximise la marge tout en assurant la classification correcte des exemples d'entraînement ;

• **Recherche des vecteurs de support :** Les exemples d'entraînement situés à proximité de l'hyperplan sont appelés des vecteurs de support. Ces vecteurs jouent un rôle crucial dans la détermination de l'hyperplan ;

• **Calcul de l'hyperplan :** l'hyperplan est calculé à l'aide des vecteurs de support. Les poids **a** et le biais **b** sont déterminés pour définir l'hyperplan de manière optimale ;

• **Évaluation du modèle :** Évaluer les performances du modèle SVM en utilisant un ensemble de données de test distinct.

II.9.3. Naïve Bayésienne (Naïve Bayes (NB))

II.9.3.1. Définition

Naïve Bayes Classifier (NB), est l'un des algorithmes cruciaux de l'apprentissage automatique qui aide à résoudre les problèmes de classification. Il est dérivé de la théorie des probabilités de Bayes et est utilisé pour l'analyse sentimentale, la classification des nouveaux articles et la filtration du spam.

L'objectif est pour classer les nouvelles observations dans des classes prédéfinies pour les données non initiées **[24]**.

II.9.3.2. Processus d'entraînement de naïve bayes

• **Importation des bibliothèques de base :** importer les bibliothèques de base requises ;

• **Importation de l'ensemble de données :** importer l'ensemble de données requis ;

• **Prétraitement des données :** Ensemble de données divisé en ensembles de données d'entraînement et test ;

• **Entraînement du modèle :** Ajustement de l'algorithme Naïve Bayes à l'ensemble de données de formation ;

• **Test et évaluation du modèle :** Prédiction des résultats de l'ensemble de données test;

• **Visualisation du modèle :** Visualiser les résultats de l'ensemble de données de test.

II.9.4. Méthodes d'ensemble (Ensemble Learning)

L'idée centrale derrière les méthodes d'ensemble (EL) est de capitaliser sur les différents classifieurs en les apprenant collectivement. Étant donné que chaque classifieur présente des avantages et des limites, certains étant performants pour détecter un type spécifique d'attaques tandis que d'autres peuvent montrer des performances médiocres sur d'autres types, l'approche d'ensemble vise à fusionner les résultats de plusieurs modèles faibles pour former un seul apprenant plus robuste. Le regroupement des modèles pour améliorer les performances peut être réalisée de diverses manières, en utilisant différents algorithmes de regroupement **[22]**, telles que :

• **Bagging :** consiste à créer plusieurs répliques d'un même modèle, en entraînant chaque modèle sur un échantillon aléatoire du jeu de données ;

• **Boosting :** consiste à entraîner successivement plusieurs modèles relativement faibles, en demandant à chaque modèle de tenter de corriger les erreurs commises par son prédécesseur ;

• **Stacking :** Plutôt que de simplement agréger les résultats des modèles pour obtenir une prédiction majoritaire, il demande à un dernier estimateur d'apprendre à distinguer les bonnes et les mauvaises prédictions d'ensemble, afin de produire lui-même la prédiction finale.

II.9.5. K-plus proches voisins (NearestNeighbor (KNN))

II.9.5.1. Définition

Le K-plus proches voisins (KNN), représente l'un des algorithmes supervisés les plus élémentaires en Machine Learning. Il se base sur le concept de "similarité des caractéristiques" pour prédire la classe d'un certain échantillon de données.

Pour ce faire, il identifie un échantillon en calculant sa distance par rapport à la base de ses voisins. Habituellement, la distance euclidienne est utilisée (représentant la ligne droite entre deux points), mais d'autres méthodes telles que la distance de Manhattan ou la distance cosinus peuvent également être utilisées **[22]**.

II.9.5.2. Modèle K-plus proches voisins

Pour bien comprendre le modelé KNN, voici les cinq étapes élémentaires :

• **Transformation des données :** convertir les données en modèle de vecteurs d'espace;

• **Entraînement du modèle :** Entraîner le KNN en utilisant l'ensemble de données d'entraînement, où le modèle stocke les vecteurs de caractéristiques et les étiquettes des échantillons associées ;

• **Calcul de la distance :** le calcul de la distance entre l'échantillon de test x et tous les autres échantillons ;

• **Tri des distances :** faire le tri des distances calculées dans l'ordre croissant.

• **Classification :** identifier les k instances les plus proches de x et sélectionner la classe majoritaire.

II.9.6. Régression Logistique (LogisticRegression (LR))

II.9.6.1. Définition

Régression logistique (LR), est un modèle de classification linéaire et d'analyse statistique qui permet d'étudier les relations entre un ensemble de variables prédictives nommées X et une variable binomiale nommée Y.

Le modèle de régression logistique utilise également l'optimisation des coefficients pour prédire la probabilité qu'un évènement soit susceptible de se produire ou non **[25]**.

II.9.6.2. Modèle Régression Logistique

• **Description des données :** La première étape consiste à explorer les données du dataset ;

• **Préparation des données :** Les données sont sauvegardées sous format CSV ;

• **Entrainement du modèle :** instancier un nouveau régresseur et l'entraine ;

• **Evaluation du modèle :** prédiction des classes d'un ensemble d'échantillons ;

• **Persistance du modèle :** Après avoir entrainé un modèle, il est souhaitable de le conserver pour un usage ultérieur.

II.10. Type d'algorithme non supervisé

II.10.1. K-Mean clustering (K-Means)

II.10.1.1. Définition

K-Mean clustering représente l'un des algorithmes non supervisés le plus populaire pour les problèmes de clustering, en plaçant les données hautement similaires dans le même cluster. L'algorithme K-Mean est l'un des algorithmes de ML itératifs populaires basés sur les centroïdes qui apprennent de manière non supervisée. K représente le nombre de centroïdes (centre du cluster) dans un ensemble de données **[25]**.

L'algorithme fonctionne en 2 étapes répétées en boucle : mais avant il faut placer au hasard un nombre K de centres dans le nuage de points, ensuite :

- L'étape 1 : consiste à rallier chaque exemple au centre le plus proche ;

- L'étape 2 : consiste à déplacer les centres au milieu de leur cluster ;

- Répète ainsi les étapes 1 et 2 en boucle jusqu'à ce que les centres ne bougent plus.

II.10.1.2. Modèle k-means

Les cinq étapes élémentaires du modèle k-means :

• **Choix du nombre de clusters (K) :** Déterminer le nombre de clusters (K) à identifier dans les données ;

• **Initialisation des centres de cluster :** Sélectionner K points initiaux (peuvent être choisis de manière aléatoire ou avec une méthode de sélection plus sophistiquée) comme centres de cluster ;

• **Affectation des paquets aux clusters :** Calculer la distance entre chaque observation et les centres de cluster.

• **Assignation des observations :** Assigner chaque observation au cluster dont le centre est le plus proche en fonction de la distance calculée ;

• **Mise à jour des centres de cluster :** Calculer les nouveaux centres de cluster en prenant la moyenne des observations attribuées à chaque cluster ;

• **Répétition des étapes 3 et 4 :** Répéter les étapes 4 et 5 jusqu'à ce qu'une condition d'arrêt soit atteinte ;

• **Interprétation des clusters :** Une fois que l'algorithme K-means a convergé, une analyse doit être effectuée pour les clusters formés afin de comprendre leur signification. Chaque cluster peut représenter un type similaire de trafic réseau.

II.11. Quel algorithme choisir ?

Le choix d'une technique appropriée à la tâche à traiter peut-être l'une des parties les plus difficiles. Pour toute tâche donnée, il existe des dizaines de techniques qui pourraient être des choix valables, et déterminer celle qui est optimale peut ne pas être évident. En général, voici quelques questions à poser lors de la sélection d'un algorithme d'IA **[25]** :

1. Quelle est la taille de votre jeu de données ?
2. Prédisez-vous une catégorie d'échantillon ou une valeur quantitative ?
3. Avez-vous des données étiquetées ?
4. De quelle quantité de données étiquetées disposez-vous ?
5. Connaissez-vous le nombre de catégories des résultats ?
6. De combien de temps et de ressources disposez-vous pour entraîner le modèle ?
7. De combien de temps et de ressources disposez-vous pour faire des prédictions ?

II.12. Comment fonctionne un NIDS basés sur l'IA

Un Système de Détection d'Intrusion du Réseau (NIDS) construit à l'aide des techniques de Machine Learning suit généralement trois principales étapes : la préparation des données, l'apprentissage, et les tests. (Voir la Figure II.3)

1. Phase de prétraitement des données : Les données sont d'abord prétraitées pour les adapter à l'algorithme sélectionné. Ensuite, elles sont divisées de manière aléatoire en deux ensembles distincts : l'ensemble de données d'entraînement et l'ensemble de données de test.

2. Phase d'apprentissage : L'algorithme de Machine Learning est ensuite entraîné en utilisant l'ensemble de données d'entraînement. Le temps d'entraînement dépend à la fois de la taille du jeu de données et de la complexité du modèle choisi.

3. Phase de test : Une fois le modèle entraîné, il est testé à l'aide de l'ensemble de données de test pour évaluer ses performances. Dans le contexte des modèles NIDS, chaque instance de trafic réseau est prédite comme étant soit normale, soit une attaque.

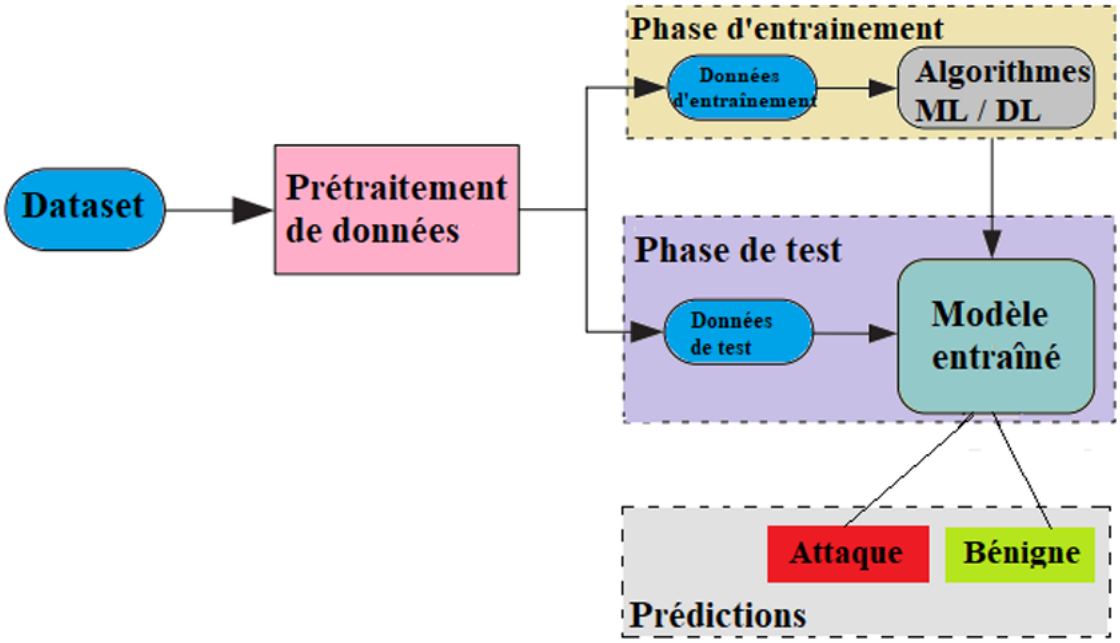

Figure II.3 : Fonctionnement des NIDS basés sur ML **[20]**.

II.13. Conclusion

En conclusion, les méthodes de l'intelligence artificielle offrent des solutions prometteuses pour la détection et la prévention des attaques réseau. Grâce à des approches telles que l'apprentissage supervisé et non supervisé, ils disposent d'outils puissants pour identifier les activités malveillantes et renforcer la sécurité des réseaux informatiques. Cependant, il est crucial de rester vigilants face à l'évolution constante des techniques d'attaques et de continuer à développer les capacités de défense pour protéger efficacement les systèmes contre les menaces en constante évolution. Le prochain chapitre état de l'art abordera ces aspects en fournissant des connaissances approfondies en s'appuyant sur des différents articles.

Chapitre III

Etat de l'art

III.1. Introduction

Ce chapitre présente un état de l'art de quelques techniques existantes dans le domaine de la détection d'intrusion utilisant l'apprentissage automatique. Nous explorerons les différentes approches, méthodes et algorithmes de l'apprentissage automatique appliqués à la détection d'intrusion, mettant en lumière les avantages, les limitations et les défis associés à chaque méthode.

III.2. Intrusion Detection System Using PCA with Random Forest Approach [26]

III.2.1.Domaine du problème

Le problème abordé dans l'article est la nécessité d'une détection précise et efficace des intrusions dans les systèmes vulnérables aux activité malveillantes sur internet. Plus précisément, l'accent est mis sur l'amélioration de la précision et de l'efficacité de l'IDS grâce à l'utilisation d'algorithmes d'apprentissage automatique tels que PCA, Random Forest, SVM, Naïve Bayes et des modèles d'apprentissage profond. Les auteurs discutent de l'application de ces méthodes dans le contexte d'amélioration des mesures de sécurité dans les environnements de communication sans fil et de la classification du trafic IoT.

III.2.2 Solution proposée

L'approche proposée utilise l'analyse en composantes principales (PCA) et la forêt aléatoire pour développer un système de détection d'intrusion (IDS) efficace pour détecter et prévenir les attaques dans les systèmes de communication sans fil.

L'analyse en composantes principales est une méthode qui réduit les dimensions des données. Elle réduit l'aspect de l'ensemble de données en un nombre souhaité d'attributs appelés composantes principales. Cette méthode prend toutes les données d'entrée comme ensemble de données de sorte que sa dimension soit très élevée puis elle réduit sa taille par déplacement des points de données sur un seul axe qui était pris sur le même axe, et les composantes principales sont exécutées.

L'algorithme de foret aléatoire est un algorithme de classification supervisée utilisée dans l'apprentissage automatique pour résoudre les problèmes de classification. Elle fonctionne en deux étapes, la première étape consiste à créer une forêt à partir de l'ensemble de données donnée en sélectionnant un sous-ensemble d'entités. Et la deuxième étape consiste à l'analyse de l'ensemble de données, c'est faire des prédictions à l'aide du classificateur crée à l'étape initial.

III.3. DDoS attack detection in software defined networking controller using machine learning techniques [27]

III.3.1.Domaine du problème

Cette recherche aborde les défis de la détection des attaques par déni de service distribué (DDoS) dans les réseaux définis par logiciel SDN (Software Defined Network) en utilisant des techniques d'apprentissage automatique. Les réseaux SDN (Software Defined Network) sont vulnérables à des menaces de sécurité pouvant compromettre les performances du réseau, ce qui nécessite des mécanismes de détection efficaces. De plus, cette recherche vise à renforcer les capacités des contrôleurs SDN (Software Defined Network) pour la détection et la gestion des attaques déni de service distribué (DDoS), assurant l'intégrité et la disponibilité des services réseau.

III.3.2.Solution proposée

La solution proposée utilise des algorithmes d'apprentissage automatique, en particulier la régression logistique (LR), Naïve Bayes (NB) et l'arbre de décision (DT), pour détecter les attaques par déni de service distribué (DDoS) dans les contrôleurs SDN (Software Defined Networking).

L'approche proposée met en correspondance en temps réel des demandes entrantes des nœuds avec un classificateur entraîné basé sur un ensemble de données spécifique au SDN généré à l'aide de l'émulateur Mininet pour la classification du trafic par apprentissage automatique. Elle compare les adresses IP source, de destination et MAC dans les requêtes entrantes avec les valeurs autorisées ; si une correspondance est trouvée, le trafic est classé comme normal, sinon, il est classé comme une attaque.

Cette méthode s'appuie sur le modèle entraîné pour différencier le trafic normal du trafic d'attaque en fonction du comportement enregistré dans le classificateur, permettant ainsi une détection efficace des attaques DDoS dans les environnements SDN.

III.4. Ensemble Machine Learning Techniques for Attack Prediction in NIDS Environment [28]

III.4.1.Domaine du problème

L'article traite de l'importance cruciale de développer des systèmes robustes de détection des intrusions réseau (NIDS) pour faire face à la menace grandissante des cyberattaques. Les systèmes actuels rencontrent des défis liés à la configuration, à la surveillance périodique et à la détection précise des intrusions. En outre, l'évolution des attaques nécessite des approches plus avancées, telles que l'utilisation de techniques d'apprentissage automatique en ensemble, pour améliorer la détection et la prévention des intrusions.

III.4.2.Solution proposée

La solution consiste à combiner une approche de filtrage avec des techniques de synchronisation d'apprentissage automatique pour sélectionner les fonctionnalités les plus appropriées pour la détection des IDS. L'objectif est de réduire le nombre de fonctionnalités tout en maintenant un taux de détection élevé des intrusions. De plus, l'utilisation des techniques d'apprentissage automatique en ensemble est explorée pour améliorer la classification des intrusions.

Le système proposé combine l'extraction, le nettoyage, l'étiquetage, la sélection et la classification des données.

Les données proviennent de la base de données du trafic réseau canadien NSL-KDD qui corrige certaines vulnérabilités et déséquilibres présents dans la version originale de la base de données KDD CUP 99, composée de cinq catégories d'attaques (R2L, normal, U2R, sonde et DoS).

Le nettoyage des données supprime les entités à valeurs fixes sur les 43 entités extraites de la base de données NSL-KDD à l'aide langage de programmation Python et 40 entités sont utilisée tandis qu'une entité est désigné comme la classe cible.

L'étiquetage des données inclut la modification de caractéristiques non numériques telles que le type de protocole et la classe en valeurs numériques après leur attribution de noms.

La sélection des données de caractéristique, la valeur probable des caractéristiques utilisé par l'approche ScikitLearn est prise en compte. Cette valeur est utilisée pour sélectionner des caractéristiques statistiquement significatives pour l'identification des irrégularités du trafic réseau.

La classification utilise les méthodes d'ensemble (apprentissage automatique en ensemble). Parmi eux :

- La méthode Naïve Bayes (NB) repose sur l'indépendance entre les caractéristiques.
- La méthode de Forêt aléatoire (RF) utilisé pour les taches de classification et de régression, Pour la classification, il utilise un système de vote pour sélectionner la classe la plus fréquente parmi les prédictions de tous les arbres. Pour la régression, il prend la moyenne des prédictions de chaque arbre.
- La méthode des k plus proches voisins (KNN) attribue une classe en fonction des points de données les plus proches.
- La méthode des arbres de décision (DT) divise les populations en fonction de prédictions importantes.

III.5. Détection d'intrusion réseau à l'aide d'une technique de sur échantillonnage et d'algorithmes d'apprentissage automatique [29]

III.5.1.Domaine du problème

Le problème abordé dans l'article est la détection des intrusions dans les réseaux informatiques, en particulier la détection des attaques. L'article mentionne que les systèmes de détection d'intrusion dans les réseaux (NIDS) actuels ne parviennent pas à détecter les menaces émergentes en raison de l'évolution des attaques modernes. De plus, le problème de déséquilibre de classe dans les ensembles de données utilisés pour former les NIDS peut affecter les performances de détection des attaques.

III.5.2. La solution proposée

La méthodologie proposée utilise l'ensemble de données UNSW-NB15 qui permet de détecter toutes les catégories d'attaques. Cette méthode commence par la technique de prétraitement des données qui implique la normalisation et la standardisation des données pour les rendre cohérentes et comparables.

Ensuite, la sélection de fonctionnalité implique la sélection des caractéristiques les plus pertinentes dans l'ensemble de données. Différentes techniques de sélection de caractéristiques telles que l'Analyse en Composantes Principales (PCA), l'Information Gain (IG) et l'Analyse de Corrélation (CA) sont utilisées pour identifier les caractéristiques les plus significatives qui contribuent à la détection des attaques.

L'Analyse de Corrélation (CA) mesure la relation entre chaque caractéristique et la variable cible, sélectionnant celles avec des valeurs proches de 1 ou -1.

L'Information Gain (IG) identifie les caractéristiques pertinentes en minimisant le bruit des caractéristiques non liées.

L'Analyse en Composantes Principales (PCA) réduit la taille des données en conservant les caractéristiques pertinentes pour la classe cible.

La méthodologie propose l'utilisation de la technique de sur échantillonnage synthétique (SMOTE) pour traiter le déséquilibre de classe dans les ensembles de données. Cette technique génère des instances synthétiques de la classe minoritaire pour équilibrer les classes et améliorer les performances de détection des attaques dans les classes sous-représentées.

Des algorithmes de classification d'apprentissage automatique sont utilisés pour la détection des attaques : Foret aléatoire, arbre de décision, régression logistique et k plus proches voisins.

III.6. Intrusion Detection Models Using Supervised and Unsupervised Algorithms - A Comparative Estimation [30]

Dans cet article une étude comparative été effectué sur les modèles de détection d'intrusion à l'aide d'algorithmes supervisés et non supervisés. Les auteurs ont utilisé les ensembles de données NSL-KDD et CICIDS pour évaluer les performances des modèles.

Dans l'apprentissage automatique supervisés, les ensembles de données NSL-KDD et CICIDS sont utilisés pour la classification binaire (données normales et d'attaques). Leurs proportions de données indiquent que NSL-KDD est presque équilibré par rapport à CICIDS.

Les données sont nettoyées et normalisées dans la phase de prétraitement des données.

Les forets aléatoires sont utilisés pour la sélection des fonctionnalités en raison de leur capacité à fournir des mesures d'importance des fonctionnalités, à améliorer la performance globale grâce à l'apprentissage d'ensemble, à réduire la variance et à résister au sur apprentissage.

Les modèles Régression logistique, forêt aléatoire, arbre de décision, Bayes naïf gaussien et le K voisins les plus proches sont créés à l'aide de la sélection d'un sous-ensemble de fonctionnalités dans l'étape précédente.

Les modèles sont entraînés avec les données d'entraînement, puis testés avec les données de test pour évaluer les performances. L'évaluation des modèles de prédictions se fait en comparant les mesures de performance (la matrice de confusion, le score F1, la précision, le rappel, l'aire sous la courbe ROC et l'exactitude).

La comparaison des algorithmes supervisés dans l'ensemble NSL-KDD et l'ensemble CICIDS montre que la forêt aléatoire a de meilleures performances que les autres algorithmes avec une précision élevée.

Dans l'apprentissage automatique non supervisés, un traitement des données est effectué comme dans l'apprentissage automatique supervisés. Les modèles K-means, forêt d'isolement et facteur aberrant local sont sélectionnés pour l'identification des clusters et la détection des anomalies.

La comparaison des algorithmes non supervisés dans l'ensemble NSL-KDD et l'ensemble CICIDS montre que le k-means a une meilleure performance que les autres algorithmes avec une précision élevée.

III.7. Synthèse :

- Dans **[30]**, L'algorithme du la foret aléatoire surpasse les autres approches d'apprentissage supervisé, prouvant son efficacité dans l'identification des anomalies, comme indiqué dans **[26]**, **[28]** et **[29]**.
- Les indicateurs de performance comme la matrice de confusion, le score F1, la précision, le rappel et l'exactitude fournissent une évaluation approfondie des modèles. Dans le cas de **[27]**, ces mesures ont démontré des performances supérieures de l'algorithme d'arbre de décision pour identifier efficacement les attaques DDoS tout en minimisant les fausses alertes.
- L'utilisation de techniques de sélection de caractéristiques telles que foret aléatoire a permis de réduire la dimensionnalité des données tout en conservant les caractéristiques les plus informatives.
- L'équilibrage des données avec des techniques comme le sous-échantillonnage améliore les performances du modèle.
- Les algorithmes d'apprentissage non supervisé comme K-Means excellent dans le clustering et la détection d'anomalies, démontrant leur utilité pour la sécurité des réseaux.
- Les modèles supervisés ont généralement surpassé les modèles non supervisés en termes de précision et de rappel.
- Dans **[26]**, L'approche utilisant PCA et forêt aléatoire offre une détection des intrusions avec une précision élevée et de faibles taux d'erreur, surpassant les techniques comme SVM, Naïve Bayes et arbre de décision.
- La PCA aide à organiser l'ensemble de données en réduisant sa dimensionnalité, ce qui peut améliorer la qualité du l'ensemble de données en conservant les attributs essentiels, le même utilisé dans **[29]**.
- La combinaison du PCA et de forêt aléatoire améliore significativement les taux de détection et réduit les fausses alertes dans les systèmes IDS.
- Dans **[27]**, Le système proposé utilisant l'algorithme arbre de décision a démontré une haute précision dans la détection des attaques DDoS sur les réseaux SDN, prouvant sa capacité à identifier les modèles de trafic malveillants avec une grande exactitude.
- La mise en œuvre d'algorithme d'arbre décision peut nécessiter des ressources de calcul importantes pour l'entraînement et l'exécution, impactant potentiellement les performances des contrôleurs SDN et le fonctionnement du réseau.
- Le déploiement et la maintenance de ces modèles dans des environnements SDN peuvent être complexes, nécessitant une expertise spécialisée.
- Il existe un risque de sur ajustement du modèle aux données d'entraînement, ce qui pourrait réduire sa capacité à généraliser et à détecter de nouveaux modèles d'attaques DDoS si le modèle n'est pas régulièrement mis à jour et ré entraîné.
- Dans **[28]**, Les performances des divers classificateurs apprentissage automatique sur la base de données NSL-KDD confirment l'efficacité des techniques d'ensemble pour renforcer les capacités de détection du NIDS.

- La mise en œuvre de méthodes foret aléatoire peut compliquer la conception et le déploiement du système, nécessitant des connaissances et des ressources spécialisées.
- L'amélioration de la précision de détection obtenue peut entraîner une surcharge de calcul, impactant les performances du système et l'utilisation des ressources.
- Bien que les performances soient améliorées sur la base de données NSL-KDD, il peut être difficile de généraliser ces résultats à divers scénarios et ensembles de données du monde réel, nécessitant une validation et des tests supplémentaires.
- Dans **[29]**, La technique SMOTE a résolu le problème de déséquilibre des classes, améliorant la précision et les performances globales de détection.
- L'algorithme de la régression logistique a montré des performances inférieures malgré la résolution du déséquilibre de classe, révélant leurs limitations pour la détection des attaques dans ce contexte.
- Bien que la sélection de caractéristiques ait amélioré les performances des modèles, son impact global sur l'exactitude des modèles reste limité.

III.8. Conclusion

Ces travaux ont montré que Les modèles d'apprentissage automatique, comme les forêts aléatoires et k-moyenne, offrent une grande précision et efficacité dans l'identification des anomalies et des intrusions. Ces techniques permettent aux systèmes de s'adapter continuellement aux nouvelles menaces grâce à l'apprentissage continu, tout en réduisant les faux positifs et en permettant une détection en temps réel. Toutefois, des défis subsistent, notamment la vulnérabilité aux attaques adversaires et le besoin de grandes quantités de données annotées. En somme, l'apprentissage automatique constitue une approche prometteuse pour améliorer la sécurité des réseaux face aux menaces évolutives.

Chapitre IV

Contribution

IV.1. Introduction

De nombreux projets et recherches ont été menés pour développer des systèmes de détection d'intrusion plus efficaces contre tous types d'attaques. Dans le chapitre précédent, nous avons évoqué les différentes méthodes appliquées avec succès à divers jeux de données spécifiques aux attaques réseau. La qualité, la performance et la précision de ces systèmes de détection dépendent largement des jeux de données utilisés. Dans ce chapitre nous allons évaluer les différents algorithmes de machine Learning afin de développer un modèle prédictif, essentiellement un classificateur, qui a la capacité de différencier les connexions malveillantes des connexions légitimes.

IV.2. Contributions

Notre code inclut plusieurs améliorations et optimisations par rapport aux versions de base des algorithmes de classification. Voici une comparaison détaillée, mettant en évidence nos contributions principales :

IV.2.1. Prétraitement des Données

• **Encodage des Variables Catégorielles :** nous avons utilisées **LabelEncoder** pour transformer les colonnes catégorielles en valeurs numériques (une étape importante de prétraitement pour les algorithmes de machine Learning).

• **Suppression des Colonnes :** nous avons supprimées la colonne **num_outbound_cmds**, parce qu'elle n'ajoute pas de valeur significative au modèle.

• **Gestion des Données Manquantes :** nous avons identifiées les colonnes avec des valeurs manquantes et leur pourcentage.

IV.2.2. Sélection des Caractéristiques

• **RFE (Recursive Feature Elimination) :** Pour améliore la pertinence des données utilisées pour l'entraînement du modèle, réduisant ainsi le bruit et les calculs inutiles, nous avons sélectionnée les 10 meilleures caractéristiques à l'aide de RFE avec **RandomForestClassifier**.

IV.2.3. Normalisation des Données

• **StandardScaler :** Pour normaliser les données, nous avons utilisées **StandardScaler** (la normalisation est cruciale pour les algorithmes sensibles à l'échelle des données).

IV.2.4. Optimisation des Hyper paramètres avec Optuna

• **Régression Logistique (Logistic Regression (LR)) :** nous avons utilisées **LogisticRegression** avec un nombre d'itérations très élevé (max_iter = 1200000), ce qui garantit une convergence.

• **K-plus proches voisins (Nearest Neighbor (KNN)) :** nous avons utilisées **Optuna** pour optimiser le nombre de voisins (n_neighbors). Cela améliore la précision de KNN en choisissant la meilleure valeur pour ce paramètre.

• **Arbre de décision (Decision Tree (DT)) :** nous avons optimisées des hyper-paramétrés **max_depth** et **max_features** pour **DecisionTreeClassifier** en utilisant **Optuna**. Ces paramètres contrôlent la complexité de l'arbre et peuvent aider à éviter le sur-apprentissage.

IV.2.5. Évaluation et Comparaison des Modèles

• **Validation Croisée :** nous avons utilisées la validation croisée (aide à évaluer la robustesse des modèles) (cross_val_score) pour évaluer les modèles avec des scores de précision et de rappel.

• **Visualisation des Scores :** nous avons créés des graphiques pour comparer les scores de précision et de rappel des différents modèles.

• **Rapports de Classification et Matrices de Confusion :** nous avons affichages des matrices de confusion et des rapports de classification pour chaque modèle. Cela fournit une vue détaillée des performances des modèles sur les données de test.

• **F1-score :** nous avons calculées et visualisées des F1-scores pour évaluer la performance globale des modèles en termes de précision et de rappel.

IV.3. Contribution clé

IV.3.1. Prétraitement et Nettoyage des Données

• Encodage des catégories.

• Gestion des valeurs manquantes.

• Normalisation des données.

IV.3.2. Sélection et Réduction des Caractéristiques

• Utilisation de RFE pour réduire le nombre de caractéristiques à celles qui sont les plus pertinentes.

IV.3.3. Optimisation des Hyper paramètres

• Utilisation d'Optuna pour trouver les meilleurs hyper-paramètres pour plusieurs modèles.

IV.3.4. Évaluation Robuste des Modèles

• Validation croisée pour obtenir une évaluation plus fiable des performances des modèles.

• Visualisation et analyse détaillée des scores de performance des modèles.

IV.3.5. Synthèse

Ces améliorations permettent d'obtenir des modèles plus précis, robustes et performants par rapport à une approche de base.

IV.4. Algorithmes de détections basés sur machine Learning

IV.4.1. Forêt aléatoire

On a commencé par créer et entraîner un classificateur de forêt aléatoire en utilisant L'algorithme de Sélection Récursive par Élimination (RFE - Recursive Feature Elimination) pour sélectionner les 10 caractéristiques les plus importantes. Ensuite, on a affiché les noms de ces caractéristiques sélectionnées. Voici un résumé de l'ensemble du processus :

1. Création d'un classificateur de forêt aléatoire ;
2. Création d'une instance RFE pour sélectionner 10 caractéristiques ;
3. Ajustement du modèle RFE sur les données d'entraînement ;
4. Création d'une carte des caractéristiques sélectionnées ;
5. Filtrage des caractéristiques sélectionnées ;
6. Affichage des caractéristiques sélectionnées ;

IV.4.2. Régression Logistique (Logistic Regression (LR))

Le code suit une structure logique pour entraîner, évaluer et optimiser un modèle de régression logistique, tout en mesurant les temps d'exécution des étapes critiques.

1. Entraînement du Modèle : La première partie du code crée une instance de LogisticRegression avec un grand nombre maximal d'itérations pour garantir la convergence. Le temps d'entraînement est mesuré en utilisant time.time() avant et après l'appel à fit(), puis le temps écoulé est affiché.

2. Prédiction : Ensuite, le code mesure le temps nécessaire pour faire des prédictions sur les données d'entraînement. Cela aide à comprendre combien de temps prend l'inférence du modèle.

3. Nouveau Modèle : Un nouveau modèle de régression logistique est créé avec une graine aléatoire fixée (random_state=42) pour assurer la reproductibilité des résultats. Ce modèle est entraîné sur les mêmes données.

4. Évaluation des Performances : Les scores d'entraînement et de test du nouveau modèle sont calculés et affichés. Le score représente la précision du modèle sur les ensembles de données d'entraînement et de test.

5. Réglage de la Verbosité d'Optuna : La dernière partie du code réduit la verbosité des logs générés par Optuna, une bibliothèque utilisée pour l'optimisation hyper paramétrique. Cela permet de minimiser les sorties de log pendant l'exécution des scripts Optuna.

IV.4.3. K-plus proches voisins (Nearest Neighbor (KNN))

1. Fonction Objective : La fonction objective définit l'objectif d'optimisation pour Optuna. Elle suggère un nombre de voisins (n_neighbors) entre 2 et 16, puis entraîne et évalue un modèle KNN avec ce paramètre sur les données d'entraînement et de test, respectivement.

2. Création et Exécution de l'Etude : study_KNN est une étude Optuna qui cherche à maximiser la précision du modèle. study_KNN.optimize(objective, n_trials=1) lance l'optimisation avec un seul essai (vous pouvez augmenter le nombre d'essais pour une meilleure optimisation).

3. Entraînement avec le Meilleur Hyperparamètre : Une fois l'optimisation terminée, le meilleur nombre de voisins (n_neighbors) est utilisé pour créer et entraîner un nouveau modèle KNN. Les scores sur les données d'entraînement et de test sont ensuite calculés et affichés.

IV.4.4. Arbre de décision (Decision Tree (DT))

1. DecisionTreeClassifier : Le modèle DecisionTreeClassifier est créé en utilisant les meilleurs hyperparamètres trouvés par Optuna : max_features et max_depth.

2. Entraînement : Le modèle est entraîné sur les données d'entraînement (x_train et y_train) en appelant la méthode fit.

3. Évaluation : Les performances du modèle sont évaluées sur les ensembles de données d'entraînement et de test en utilisant la méthode score, qui renvoie la précision du modèle.

4.Scores : sont ensuite affichés pour voir la performance du modèle sur les deux ensembles de données.

IV.5. Outils et environnement de développement

Afin d'implémenter et tester nos modèles, On a commencé par la configuration d'un environnement local de développement Python (Jupyter) utilisant la plate-forme Anaconda.

• **Python :** est le langage de programmation que nous avons choisi. C'est un langage interprété, multi-paradigme et multi-plateforme. De plus, il est très couramment utilisé et populaire pour l'apprentissage automatique et l'intelligence artificielle en raison de sa flexibilité et de la disponibilité d'un grand nombre de bibliothèques logicielles open source. Parmi les bibliothèques utilisées dans notre projet, on trouve notamment Pandas, Numpy et Scikit-Learn… etc **[34]**.

• **Anaconda :** est un gestionnaire de paquets, un gestionnaire d'environnement, une distribution de données scientifiques pour Python/R et une collection de plus de 7 500 paquets open-source **[35]**.

• **Spyder :** est un environnement scientifique puissant écrit en Python, pour Python. Il combine des fonctionnalités avancées d'édition, d'analyse, de débogage et de profilage d'un outil de développement complet avec des capacités d'exploration de données, d'exécution interactive, d'inspection approfondie et de visualisation d'un ensemble scientifique **[36]**.

• **Jupyter Notebook :** est une application Web à code source ouvert qui offre la possibilité de créer et de partager des documents comprenant du code en direct, des équations et des visualisations. Ses applications couvrent divers domaines tels que le nettoyage et la transformation des données, la simulation numérique, la modélisation statistique, la visualisation des données, et même l'apprentissage automatique **[37]**.

IV.6. Bibliothèques essentielles

IV.6.1. Importation de bibliothèques de base et de visualisation [38]

• **numpy (np) :** Bibliothèque pour le calcul scientifique avec des tableaux multidimensionnels.

• **pandas (pd) :** Bibliothèque pour la manipulation et l'analyse de données, utilisant des structures de données comme DataFrame.

• **seaborn (sns) :** Bibliothèque de visualisation basée sur matplotlib, utilisée pour rendre les graphiques plus attrayants et plus informatifs.

• **matplotlib.pyplot (plt) :** Bibliothèque de tracé pour créer des graphiques en 2D.

• time : Module pour gérer les fonctions temporelles.

• **Itertools :** Module pour créer des itérateurs efficaces.

• warnings : Module pour gérer les avertissements générés par le code.

IV.6.2. Importation des bibliothèques spécifiques à la science des données et au machine learning [39]

IV.6.2.1. Fonctions et classes de pandas

• **is_numeric_dtype :** Fonction pour vérifier si une colonne de DataFrame est de type numérique.

IV.6.2.2. Fonctions et classes de scikit-learn (sklearn)

• **tree :** Module contenant des algorithmes d'arbres de décision.

• **train_test_split :** Fonction pour diviser les données en ensembles d'entraînement et de test.

• **cross_val_score :** Fonction pour évaluer les scores des modèles en utilisant la validation croisée.

• **KNeighborsClassifier :** Algorithme des k-plus proches voisins pour la classification.

• **LogisticRegression :** Modèle de régression logistique pour la classification.

• **StandardScaler :** Classe pour standardiser les caractéristiques en supprimant la moyenne et en mettant à l'échelle à la variance unitaire.

• **LabelEncoder :** Classe pour encoder des étiquettes avec une valeur comprise entre 0 et n_classes-1.

• **DecisionTreeClassifier :** Classe pour la classification avec des arbres de décision.

• **AdaBoostClassifier :** Algorithme de boosting adaptatif pour améliorer la performance des modèles faibles.

• **VotingClassifier :** Classe pour combiner les prévisions de plusieurs modèles.

• **GradientBoostingClassifier :** Algorithme de boosting par gradient pour la classification.

• **RandomForestClassifier :** Classe pour l'algorithme de forêt aléatoire.

• **SVC :** Classe pour les machines à vecteurs de support avec noyau.

• **LinearSVC :** Classe pour les machines à vecteurs de support linéaires.

• **BernoulliNB :** Classificateur Naive Bayes pour des données binaires.

• **confusion_matrix :** Fonction pour créer une matrice de confusion.

• **classification_report :** Fonction pour générer un rapport de classification.

• **f1_score :** Fonction pour calculer le score F1, une mesure de la précision du modèle.

IV.6.2.3. Bibliothèques de machine learning spécialisées [39]

• **LGBMClassifier :** Classificateur utilisant LightGBM.

• **RFE :** Sélection de caractéristiques par élimination récursive.

• **XGBClassifier :** Classificateur utilisant XGBoost.

IV.6.2.4. Autres

• **tabulate :** Bibliothèque pour afficher les données sous forme de tableau.

VI.7. Dataset

Ce travail consiste en une évaluation des différents modèles de détection d'intrusion dans le réseau. Pour ce faire, nous avons utilisées le jeu de données pour la détection d'intrusion, NSL KDD, disponible sur **https://www.unb.ca/cic/datasets/nsl.html**.

Un environnement python est requis ainsi que certaines bibliothèques tels que numpy, pandas, matplotlib.

VI.7. 1. Choix de Dataset

La méthode la plus efficace pour évaluer les performances d'un IDS consiste à le tester à l'aide d'un jeu de données public standardisé. L'objectif est de comparer différents systèmes. Il existe peu de jeux de données publics utilisés dans la détection d'intrusions. Parmi les ensembles de données couramment utilisés, on trouve notamment KDD Cup99, NSL-KDD, l'ensemble de données UNSW-NB15, et le dataset CIC-IDS. Dans nos expérimentations, nous avons utilisé le NSL-KDD, une version améliorée de KDD99.

Cette version résout le problème de redondance des données et est l'un des jeux de données de référence les plus utilisés pour évaluer les performances de l'IDS **[40]**.

VI.7. 2. Description de la base NSL-KDD

La base NSL-KDD (Network Security Layer-Knowledge Discovery in Databases) est fondée sur l'ensemble de données KDD99. Cette dernière est une base de données contenant des connexions TCP/IP extraites de l'ensemble de données d'évaluation des systèmes de détection d'intrusions. KDD99 a été réalisée en 1998 par l'agence américaine DARPA (Défense Advanced Research Projects Agency) et L'AFRL (Laboratoire de recherche de l'armée de l'air), puis le MIT Lincoln Labs a collecté et distribué les ensembles de données pour l'évaluation des systèmes de détection d'intrusions sur les réseaux informatiques **[41]**.

La base NSL-KDD est une version réduite de KDD99, proposée en 2010 par des chercheurs dans le domaine de la détection d'intrusions réseau pour résoudre certains problèmes apparus dans la base KDD99. NSL-KDD est considérée comme un ensemble de données de référence permettant aux chercheurs de comparer différentes méthodes de détection d'intrusions. NSL-KDD présente les différences suivantes par rapport à KDD99 :

• Elle n'inclut pas les enregistrements redondants dans les données d'apprentissage, améliorant ainsi les performances de classification.

• Il n'y a pas d'enregistrements en double dans les ensembles de test proposés, ce qui contribue à obtenir de meilleurs taux de détection.

• Le nombre d'enregistrements dans les données d'apprentissage et les ensembles de test est raisonnable, ce qui permet d'exécuter les expériences sur l'ensemble complet sans avoir à choisir une petite portion au hasard. Par conséquent, les résultats d'évaluation des travaux de recherche seront cohérents et comparables.

```
Data columns (total 42 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   duration                     25192 non-null  int64
 1   protocol_type                25192 non-null  object
 2   service                      25192 non-null  object
 3   flag                         25192 non-null  object
 4   src_bytes                    25192 non-null  int64
 5   dst_bytes                    25192 non-null  int64
 6   land                         25192 non-null  int64
 7   wrong_fragment               25192 non-null  int64
 8   urgent                       25192 non-null  int64
 9   hot                          25192 non-null  int64
 10  num_failed_logins            25192 non-null  int64
 11  logged_in                    25192 non-null  int64
 12  num_compromised              25192 non-null  int64
 13  root_shell                   25192 non-null  int64
 14  su_attempted                 25192 non-null  int64
 15  num_root                     25192 non-null  int64
 16  num_file_creations           25192 non-null  int64
 17  num_shells                   25192 non-null  int64
 18  num_access_files             25192 non-null  int64
 19  num_outbound_cmds            25192 non-null  int64
 20  is_host_login                25192 non-null  int64
 21  is_guest_login               25192 non-null  int64
 22  count                        25192 non-null  int64
 23  srv_count                    25192 non-null  int64
 24  serror_rate                  25192 non-null  float64
 25  srv_serror_rate              25192 non-null  float64
 26  rerror_rate                  25192 non-null  float64
 27  srv_rerror_rate              25192 non-null  float64
 28  same_srv_rate                25192 non-null  float64
 29  diff_srv_rate                25192 non-null  float64
 30  srv_diff_host_rate           25192 non-null  float64
 31  dst_host_count               25192 non-null  int64
 32  dst_host_srv_count           25192 non-null  int64
 33  dst_host_same_srv_rate       25192 non-null  float64
 34  dst_host_diff_srv_rate       25192 non-null  float64
 35  dst_host_same_src_port_rate  25192 non-null  float64
 36  dst_host_srv_diff_host_rate  25192 non-null  float64
 37  dst_host_serror_rate         25192 non-null  float64
 38  dst_host_srv_serror_rate     25192 non-null  float64
 39  dst_host_rerror_rate         25192 non-null  float64
 40  dst_host_srv_rerror_rate     25192 non-null  float64
 41  class                        25192 non-null  object
```

Figure VI.1 : Description de la base NSL-KDD.

VI.8. Mesures d'évaluation des algorithmes

Voici les critères les plus importants utilisés pour une évaluation plus réaliste de notre travail : précision ; Rappel ; F1 score.

VI.8.1. Précision (Precision)

Lors de cette étape, nous avons calculé la métrique de précision, ce qui nous a permis d'identifier avec exactitude les intrusions avérées.

Cette mesure est essentielle pour évaluer l'efficacité de notre système de détection des intrusions en distinguant les vraies alertes des fausses.

• K-plus proches voisins (Nearest Neighbor (KNN)):

- **Score d'entraînement (Train Score): 0.978394**

Cela signifie que le modèle a une précision de 97.84 % sur les données d'entraînement. Le modèle s'adapte bien aux données sur lesquelles il a été formé.

- **Score de test (Test Score): 0.978433**

Cela montre que le modèle a une précision de 97.84 % sur les données de test. La performance sur les données de test est presque identique à celle sur les données d'entraînement, ce qui indique que le modèle généralise bien aux nouvelles données.

- **Temps d'entraînement (Training time): 0.024067 secondes**

Le temps nécessaire pour entraîner le modèle est très court, seulement environ 0.024 secondes. Cela montre que le modèle KNN est rapide à entraîner.

- **Temps de test (Testing time): 0.010076 secondes**

Le temps nécessaire pour tester le modèle est également très court, environ 0.01 secondes. Cela indique que le modèle KNN est rapide à prédire les résultats pour les nouvelles données.

• Régression Logistique (Logistic Regression (LR)):

- **Score d'entraînement (Train Score): 0.941816**

Le modèle a une précision de 94.18 % sur les données d'entraînement. Cela montre une bonne performance sur les données d'apprentissage.

- **Score de test (Test Score): 0.938872**

Le modèle a une précision de 93.89 % sur les données de test. La légère baisse par rapport au score d'entraînement indique que le modèle généralise bien mais pourrait être légèrement sur-ajusté.

- **Temps d'entraînement (Training time): 0.050286 secondes**

Le temps nécessaire pour entraîner le modèle est relativement court, environ 0.05 secondes, ce qui est rapide pour un modèle de régression logistique.

- **Temps de test (Testing time): 0.000997 secondes**

Le temps nécessaire pour tester le modèle est extrêmement court, environ 0.001 secondes, ce qui montre que le modèle est très rapide pour faire des prédictions.

• Arbre de décision (Decision Tree (DT)) :

- **Score d'entraînement (Train Score) : 1.0**

Le modèle a une précision parfaite de 100 % sur les données d'entraînement.

- **Score de test (Test Score) : 0.993384**

Le modèle a une précision de 99.34 % sur les données de test.

Model	Train Score	Test Score
KNN	0.978394	0.978433
Logistic Regression	0.941817	0.938873
Decision Tree	1	0.993384

Tableau VI.1 : Résumé de la précision des 3 algorithmes de Machine Learning.

VI.8.2. Rappel (Recall)

Dans cette partie, nous avons complété l'analyse de la précision en intégrant les événements normaux correctement classés par le système.

Comme le montre le **tableau VI.2**, le modèle d'arbre de décision a surpassé les deux autres modèles en termes de performances pour les deux métriques analysées.

Modelé	Précision	Rappel
K-plus proches voisins (Nearest Neighbor (KNN))	98.39 % +- 0.41	98.3 % +- 0.44
Régression Logistique (Logistic Regression (LR))	93.57 % +- 0.63	95.64 % +- 0.6
Arbre de décision (Decision Tree (DT))	99.39 % +- 0.28	99.44 % +- 0.15

Tableau VI.2 : Résumé de la précision et du rappel pour les trois algorithmes de Machine Learning.

VI.8.3. F1 score

Dans cette dernière partie, nous avons réalisé une prédiction complète en utilisant le score F1. Nous avons également calculé la matrice de confusion, ce qui nous a permis de regrouper et de résumer les évaluations des performances du modèle.

VI.8.3.1. K-plus proches voisins (Nearest Neighbor (KNN))

- **Matrice de confusion :**

Avec cet algorithme, nous avons détecté 57 événements anormaux comme étant normaux, ce qui indique que la précision des prédictions est modérée.

		Résultat de prédictions	
		Normal	Anormal
Evènement actuel	Normal	3423	75
	Anormal	57	4003

Tableau VI.3 : Matrice de confusion pour l'algorithme K-plus proches voisins (Nearest Neighbor (KNN)).

• Résumes des évaluations :

Dans ce résumé des évaluations, nous constatons que l'algorithme des K-plus proches voisins (K-Nearest Neighbors, KNN) fournit une prédiction précise, atteignant un taux de précision de 98 %.

	Précision	Rappel	F1-Score	Support
Normal	0.98	0.98	0.98	3498
Anormal	0.98	0.99	0.98	4060
Accuracy			0.98	7558
Macro avg	0.98	0.98	0.98	7558
weighted avg	0.98	0.98	0.98	7558

Tableau VI.4 : Résumé des évaluations pour l'algorithme K-plus proches voisins (Nearest Neighbor (KNN)).

VI.8.3.2. Régression Logistique (Logistic Regression (LR))

• Matrice de confusion :

Avec cet algorithme, nous avons détecté 188 événements anormaux comme étant normaux, ce qui représente une prédiction inférieure comparée à celle de l'algorithme des K-plus proches voisins (K-Nearest Neighbors, KNN).

		Résultat de prédictions	
		Normal	Anormal
Evènement actuel	Normal	3224	724
	Anormal	188	3872

Tableau VI.5 : Matrice de confusion pour l'algorithme Régression Logistique (Logistic Regression (LR)).

• Résumes des évaluations :

Dans ce résumé des évaluations, il est notable que l'algorithme de Régression Logistique (Logistic Regression, LR) offre une prédiction modeste, atteignant un taux de précision de 94 %.

	Précision	Rappel	F1-Score	Support
Normal	0.94	0.92	0.93	3498
Anormal	0.93	0.95	0.94	4060
Accuracy			0.94	7558
Macro avg	0.94	0.94	0.94	7558
weighted avg	0.94	0.94	0.94	7558

Tableau VI.6 : Résumé des évaluations pour l'algorithme Régression Logistique (Logistic Regression (LR)).

VI.8.3.3. Arbre de décision (Decision Tree (DT))

• Matrice de confusion :

Dans cet algorithme, nous avons identifié 20 événements anormaux ont été incorrectement classifiés comme normaux, ce qui témoigne d'une excellente prédiction comparativement aux résultats obtenus avec les méthodes K-plus proches voisins (KNN) et de Régression Logistique (LR).

		Résultat de prédictions	
		Normal	Anormal
Evènement actuel	Normal	3478	20
	Anormal	20	4040

Tableau VI.7 : Matrice de confusion pour l'algorithme Arbre de décision (Decision Tree (DT)).

• Résumes des évaluations :

Dans ce résumé des évaluations, on constate que l'Arbre de décision (Decision Tree, DT) affiche une performance très élevée avec un taux de précision de 99 %.

	Précision	Rappel	F1-Score	Support
Normal	0.99	0.99	0.99	3498
Anormal	1.00	1.00	1.00	4060
Accuracy			0.99	7558
Macro avg	0.99	0.99	0.99	7558
weighted avg	0.99	0.99	0.99	7558

Tableau VI.8 : Résumé des évaluations pour l'algorithme qu'Arbre de décision (Decision Tree (DT)).

VI.9. Résultats et comparaison

La figure IV.2 offre une visualisation claire des deux classes : la classe normale est représentée en bleu, tandis que la classe anormale (attaque) est indiquée en orange. On peut clairement observer que la classe normale prédomine largement sur la classe anormale dans cette représentation graphique. Cela souligne la répartition inégale entre les événements normaux et les événements anormaux dans l'ensemble de données analysé.

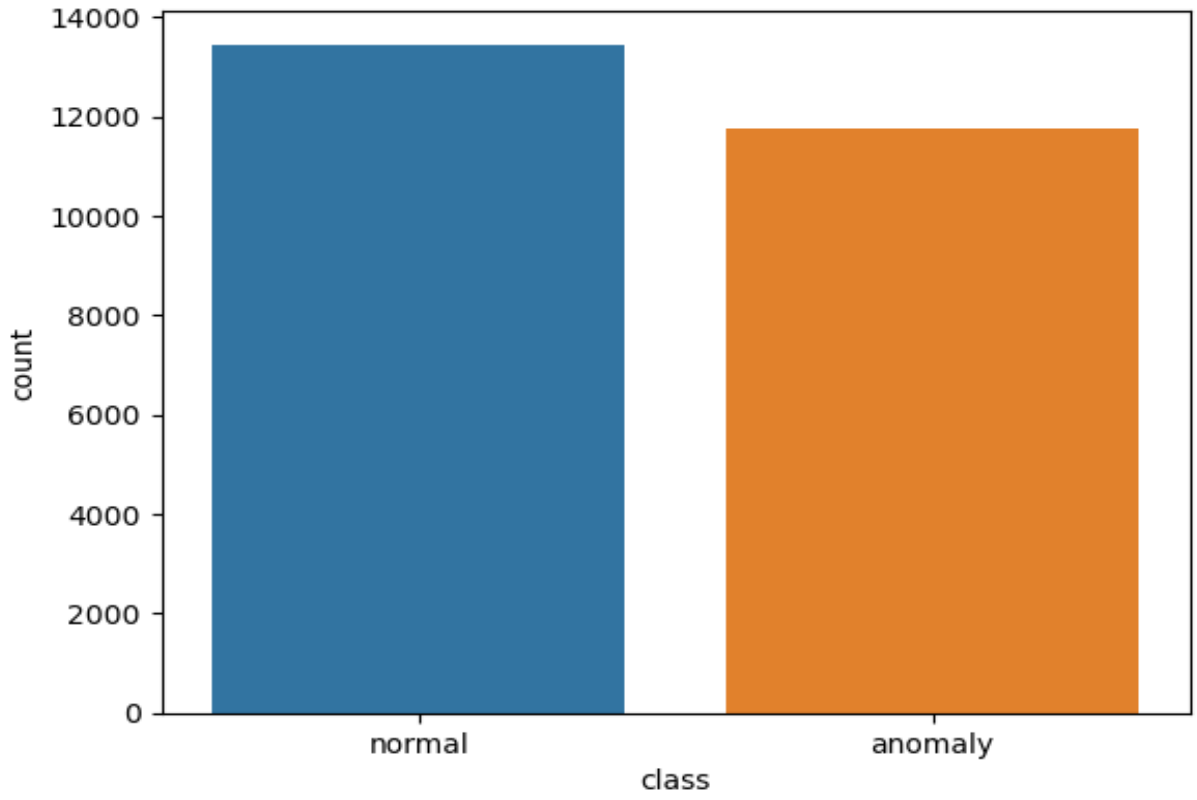

Figure VI.2 : Les catégories de classes.

La figure IV.3 présente une analyse comparative des performances des trois algorithmes, illustrant leurs résultats en termes de précision et de rappel. Il est manifeste que l'arbre de décision se distingue nettement, offrant à la fois une précision et un rappel supérieurs à ceux observés pour les deux autres algorithmes. Cela suggère que l'arbre de décision est particulièrement efficace pour identifier et classifier les données dans cette étude, surpassant ainsi ses concurrents dans les deux aspects clés de l'évaluation des modèles.

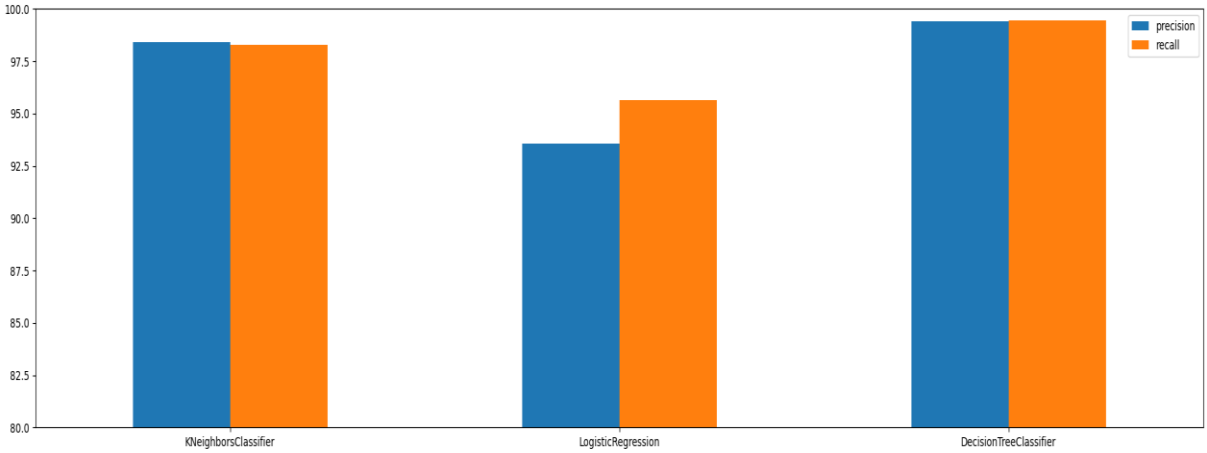

Figure VI.3 : Analyse comparative des performances des trois algorithmes.

Dans la figure IV.4, en analysant la comparaison entre les trois algorithmes à l'aide du score F1, il est clair que l'arbre de décision surpasse les autres en termes de précision et de rappel. Cette observation souligne la robustesse de l'arbre de décision dans la détection des événements, avec un taux impressionnant de 0,99 %. En conclusion, ces résultats démontrent que l'algorithme de l'arbre de décision excelle comme choix optimal pour la détection.

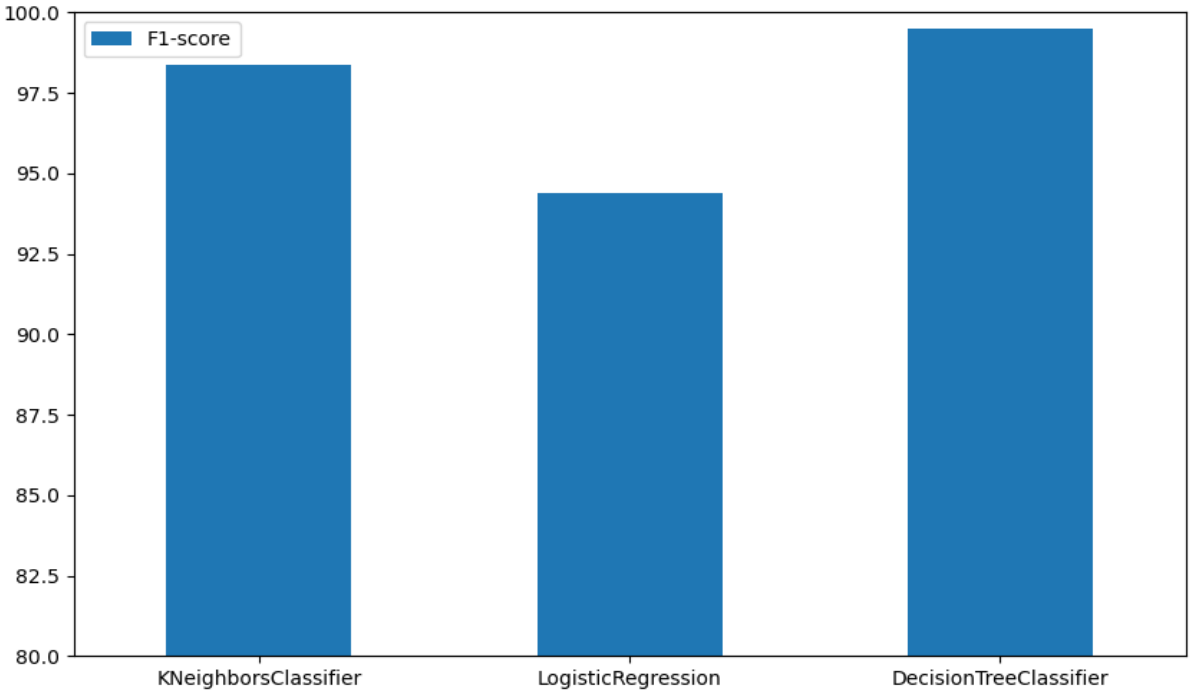

Figure VI.4 : Comparaison des trois algorithmes de Machine Learning.

Les résultats révèlent que la méthode Arbre de décision (Decision Tree (DT)) est notablement plus efficace en termes de taux de précision comparée aux méthodes du K-plus proches voisins (Nearest Neighbor (KNN)) et de la Régression Logistique (Logistic Regression (LR)).

VI.10. Conclusion

En conclusion, ce travail s'est concentré sur les techniques de détection d'intrusion en se focalisant particulièrement sur les méthodes d'apprentissage automatique. Nous avons réalisé une comparaison entre ces méthodes en utilisant le taux de précision comme critère d'évaluation sur la base de données NSL-KDD. Les résultats indiquent que la méthode de l'Arbre de décision (Decision Tree, DT) est la plus performante en termes de précision par rapport aux méthodes des K-plus proches voisins (Nearest Neighbor (KNN)) et de la Régression Logistique (Logistic Regression (LR)).

Conclusion générale et perspectives

Conclusion générale et perspectives

En conclusion, l'application de divers algorithmes d'apprentissage automatique pour la détection des intrusions sur les réseaux informatiques permet de créer des modèles prédictifs performants capables de distinguer efficacement les connexions malveillantes des connexions normales. Cette approche améliore la capacité des systèmes de détection des intrusions (IDS) à identifier les menaces, même celles provenant d'initiés, en offrant une protection renforcée contre les accès non autorisés. L'intégration de techniques avancées d'apprentissage automatique dans les IDS représente une avancée significative dans le domaine de la cyber sécurité, rendant les réseaux informatiques plus résilients face aux attaques sophistiquées.

L'état de l'art pour les techniques de détection d'intrusion utilisant l'apprentissage automatique démontre une avancée significative en termes de précision et d'efficacité, offrant des solutions robustes pour identifier et répondre aux menaces de sécurité émergentes, tout en soulignant la nécessité d'une adaptation continue aux nouvelles menaces et d'une optimisation des modèles pour des performances accrues.

Les résultats de cette recherche démontrent que l'utilisation de techniques d'apprentissage automatique, telles que les méthodes d'apprentissage supervisé, permettent d'améliorer considérablement la détection des activités malveillantes dans les réseaux informatiques.

Pendant la réalisation de notre projet de fin d'étude, nous avons eu l'opportunité d'acquérir de nombreuses compétences pratiques et théoriques. Nous avons appliqué les connaissances acquises au cours de notre formation pour développer un modèle de machine Learning, en apprenant à maîtriser le langage Python et en suivant méthodiquement les différentes étapes nécessaires à la création d'un modèle performant. Cette expérience nous a permis de comprendre en profondeur les processus de prétraitement des données, de sélection des caractéristiques, et d'optimisation des hyper paramètres. Ces compétences, combinées à notre expertise en machine Learning, sont directement transposables au domaine professionnel, notamment dans des rôles tels que data scientiste ou ingénieur en apprentissage automatique.

Pour notre part, nous espérons dans le future, étudier l'intégration d'autres techniques de la part des produits ultramodernes et estimer leurs avantages dans le domaine de la sécurité informatique.

Bibliographie

[1] Site Internet URL : https://www.crowdstrike.fr/cybersecurity-101/it-security/, « SÉCURITÉ INFORMATIQUE », consultée le 13 juillet 2022.

[2] N. LABRAOUI, « Sécurité Informatique », Cours Master 1 Réseaux et systèmes distribués, Université Abou Bakr Belkaid, 2019-2020.

[3] D. RIQUET, « Une architecture de détection d'intrusions réseau distribuée basée sur un langage dédié », Article journal, HAL open science, 4 avril 2018.

[4] Site Internet URL : https://www.ionos.fr/digitalguide/serveur/securite/ping-of-death/, consultée le 21 mars 2024.

[5] Site Internet URL : https://web.maths.unsw.edu.au/~lafaye/CCM/attaques/attaque-land.htm, consultée le 21 mars 2024.

[6] Site Internet URL : https://www.orangecyberdefense.com/fr/insights/blog/ethical-hacking/hacks-de-legende-2-1999-melissa, consultée le 21 mars 2024.

[7] Site Internet URL : https://fr.wikipedia.org/wiki/Stuxnet, consultée le 21 mars 2024.

[8] Site Internet URL : https://www.zdnet.fr/actualites/10-ans-de-malwares-en-2012-shamoon-et-flame-39895037.htm, consultée le 21 mars 2024.

[9] Site Internet URL : https://www.tf1info.fr/high-tech/yahoo-trois-milliards-de-comptes-pirates-dans-la-cyber-attaque-de-2013-2066275.html, consultée le 21 mars 2024.

[10] Site Internet URL : https://altoo.io/fr/histoire-des-cyberattaques/, consultée le 21 mars 2024.

[11] Site Internet URL : https://www.cloudflare.com/fr-fr/learning/ddos/glossary/mirai-botnet/, consultée le 21 mars 2024.

[12] Site Internet URL : https://www.air-journal.fr/2020-05-20-cyber-attaque-chez-easyjet-9-millions-de-clients-affectes-5220290.html, consultée le 21 mars 2024.

[13] Site Internet URL: https://www.lemondeinformatique.fr/actualites/lire-apres-un-ransomware-le-ch-d-arles-panse-les-plaies-de-son-si-83908.html, consultée le 21 mars 2024.

[14] Site Internet URL: https://www.lemondeinformatique.fr/actualites/lire-apres-un-ransomware-le-ch-d-arles-panse-les-plaies-de-son-si-83908.html, consultée le 21 mars 2024.

[15] E. DETOISIEN, « Les attaques externes », Article journal, Linux Focus, 2003.

[16] H. AMARIR, A. DANES, S. DOFFE, « La protection des réseaux contre les attaques DoS », Article, Université Paris Descartes, 2009.

[17] L. LEVIER, « Attaques des réseaux », Article de référence, Cartographier le réseau, 17 févr. 2020.

[18] S. HANSMA, R. HUNT, « A taxonomy of network and computer attacks », Computers and Security, 2005.

[19] Site Internet URL : https://fr.wikipedia.org/wiki/SYN_flood, consultée le 18 avril 2024.

[20] Site Internet URL : https://www.geekmaispasque.com/2019/10/intelligence-artificielle-machine-learning-deep-learning-liens-differences/?cn-reloaded=1, consultée le 15 avril 2024.

[21] Équipe Blent (Data Scientist), « L'apprentissage supervisé : définition et exemples », Article web, Catégorie Machine Learning, le 12 avril 2022.

[22] Y. BENZAKI, « Tutoriel de classification de fleurs d'IRIS avec la Régression logistique et Python », 24 octobre 2018.

[23] A. KONATE, « Un aperçu sur quelques méthodes en apprentissage automatique supervisé », Document scientifique, HAL open science, 05 décembre 2018.

[24] Ecrivain chevronné, « Une introduction à l'algorithme Naïve Bayes pour les débutants », Document scientifique, Turing, 2024.

[25] V. MATHIVET, « Machine Learning », Livre, 325 pages, octobre 2021.

[26] S. WASKLE, L. PARASHAR et U. SINGH, « Intrusion Detection System Using PCA with Random Forest Approach », Conférence internationale sur l'électronique et les systèmes de communication durables, Université de technologie d'Auckland, aout 2022. DOI : 10.1109/ICESC48915.2020.9155656.

[27] A. ALTAMIMI, A. ABDULHASSAN et N. OBEIS « DDoS attack detection in software defined networking controller using machine learning techniques », Bulletin de génie électrique et informatique, Université de Babylone, Irak, Octobre 2022. DOI : 10.11591/eei. v11i5.4155.

[28] T. SREENIVASULA et R. SATHYA, « Ensemble Machine Learning Techniques for Attack Prediction in NIDS Environment », revue, 18 Mars 2022.

[29] H. AHMED, A. HAMEED et N. BAWANY « Network intrusion detection using oversampling technique and machine learning algorithms », PeerJ Computer Science, Département d'informatique et de génie logiciel, Université Jinnah Pakistan, 7 janvier 2022. DOI: https://doi.org/10.7717/peerj-cs.820.

[30] A. SIRISHA, K. CHAINTANYA, K. KRISHNA et S. KANUMALLI, « Intrusion Detection Models Using Supervised and Unsupervised Algorithms - A Comparative Estimation », Revue internationale d'ingénierie de sûreté et de sécurité,Institut d'Information et de Technologie de Vignan Inde, 01 Fevrier 2021. DOI: https://doi.org/10.18280/ijsse.110106.

[31] Site Internet URL : https://fastercapital.com/fr/contenu/Revolutionner-la-gestion-des-reseaux-avec-la-commutation-SDN, consultée le 17 avril 2024.

[33] F. LUCINI, Les données synthétiques: vecteur de rapidité, de sécurité et de scalabilité, 6 Janvier 2022.

[34] Site Internet URL: https://www.journaldunet.fr/web-tech/dictionnaire-du webmastering/1445304-python-definition-et-utilisation-de-ce langage-informatique/, Consulté le 10 juin 2024.

[35] Site Internet URL: https ://docs.anaconda.com/anaconda/, Consultée le 10 juin 2024.

[36] Site Internet URL: https ://docs.spyder-ide.org/current/index.html, Consulté le 10 juin 2024.

[37] Site Internet URL: https ://jupyter.org/about, Consultée le 10 juin 2024.

[38] Site Internet URL: https://docs.kanaries.net/fr/articles/top10-dataviz-in-python, Consulté le 10 juin 2024.

[39] Site Internet URL: https://mobiskill.fr, Consultée le 10 juin 2024.

[40] B.Ritu et N.Ritu, revue de l'ensemble de données kdd cup99 et nsl nsl-kdd, journal de recherche avancée en informatique, 2019.

[41] Site Internet URL: https://learn.saylor.org, Consultée le 10 juin 2024.