République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université20Août1955-Skikda
Faculté des Sciences Département d'Informatique

Mémoire

En vue de l'obtention du Diplôme de Master en Informatique

Spécialité :Réseaux et Systèmes Distribués (RSD)

Intitulé :

"Détection d'Intrusions dans les Réseaux Informatiques à l'Aide de l'Apprentissage Automatique"

Réalisé par:

- FILALI Berrezoug

Devant le jury:

-Dr. SEDDARI Noureddine	Président	Universitéde20Août1955-Skikda
-Dr. MAGROUN Hanene	Examinateur	Université de20Août1955-Skikda
-Dr. BOUGAMOUZA Fateh	Encadreur	Université de20Août1955-Skikda

Année universitaire 2023/2024

DEDICACES

Je dédie ce travail

À mes chers parents,

Berrezoug,

REMERCIEMENTS

Je remercie profondément et énormément à Monsieur ***Bougamouza*** qui m'a permis de bénéficier de son encadrement , Merci pour votre patience et conseils.

Mes vifs remerciements aux membres du jury pour avoir accepté d'examiner ce travail et d'enrichir avec leur précieuses propositions.

Résumé

Durant les deux dernières décennies, le développement fulgurant des technologies de l'information et de la communication, en particulier Internet, a déclenché une vagued'innovations sans précédent à l'échelle mondiale. Cependant, cette progression s'accompagne d'une augmentation significative des cyber-attaques, mettant en péril la sécurité, la confidentialité et la disponibilité des réseaux informatiques. Les intrusions peuvent gravement perturber les opérations, engendrer des pertes financières importantes et nuire à la réputation des organisations. Face à ces menaces croissantes, la protection des réseaux de communication devient un défi majeur pour l'avenir. Les systèmes de détection d'intrusion (IDS) se révèlent essentiels en tant que première ligne de défense, ayant fait l'objet de nombreuses recherches pourleur capacité à identifier et prévenir les activités malveillantes avant qu'elles ne causent des dommages irréparables.

Ce travail présente l'état de l'art de deux domaines prometteurs de l'informatique : l'intelligence artificielle et la détection d'intrusions, avec pour objectif de développer un modèle capable de détecter et de classifier automatiquement une large gamme d'attaques. Les contributions de ce mémoire incluent l'exploration de diverses méthodes d'apprentissage automatique, l'étude approfondie de différentes techniques de l'apprentissage automatique, ainsi que l'analyse des résultats pour évaluer l'efficacité des différentes approches. Les performances de divers algorithmes d'apprentissage ont été évaluées dans le contexte de la détection d'intrusions en utilisant la base de données publique CIC-IDS2017. Après un prétraitement et une normalisation des données, ces algorithmes ont été appliqués pour la classification, et leurs performances ont été comparées à l'aide de différentes mesures d'évaluation.Les algorithmes utilisés dans cette étude incluent les arbres de décision (DT), Naive Bayes (NB), XGBoost, les machines à vecteursde support (SVM), les k-plus proches voisins (KNN) et la régression logistique (LR). Les résultats obtenus montrent que chaque algorithme présente des performances variées en termes de précision, de rappel, de mesure F1 et de taux de faux positifs. Parmi les algorithmes testés, XGBoost et les arbres de décision se sont distingués par leurs performances élevées, offrant des niveaux de précision et de rappel supérieurs par rapport aux autres méthodes. Les algorithmes KNN et SVM ont également montré de bonnes performances, bien que légèrement inférieures à celles de XGBoost et DT. La régression logistique et Naive Bayes, tout en étant moins

performants dans certains cas, ont démontré une robustesse et une rapidité d'exécution appréciables.

Ces résultats soulignent l'importance de choisir l'algorithme adapté en fonction des spécificités des données et des exigences du système de détection d'intrusions.

Mots clés : Intelligence artificielle, Système de détection d'intrusion (IDS), Cyber-attaques, Apprentissage automatique, CIC-IDS2017.

Abstract:

In the past two decades, the rapid development of information and communication technologies, especially the Internet, has triggered an unprecedented wave of innovations worldwide. However, this progress has been accompanied by a significant increase in cyberattacks, jeopardizing the security, confidentiality, and availability of computer networks. Intrusions can severely disrupt operations, cause substantial financial losses, and damage the reputation of organizations. Faced with these growing threats, protecting communication networks becomes a major challenge for the future. Intrusion Detection Systems (IDS) are essential as the first line of defense, having been the subject of extensive research for their ability to identify and prevent malicious activities before they cause irreparable damage.

This work presents the state of the art in two promising fields of computer science: artificial intelligence and intrusion detection, with the objective of developing a model capable of automatically detecting and classifying a wide range of attacks. The contributions of this thesis include the exploration of various machine learning methods, an in-depth study of different machine learning techniques, as well as the analysis of results to evaluate the effectiveness of different approaches. The performance of various learning algorithms was assessed in the context of intrusion detection using the public CIC-IDS2017 dataset. After data preprocessing and normalization, these algorithms were applied for classification, and their performance was compared using different evaluation metrics. The algorithms used in this study include decision trees (DT), Naive Bayes (NB), XGBoost, support vector machines (SVM), k-nearest neighbors (KNN), and logistic regression (LR). The results show that each algorithm exhibits varying performance in terms of accuracy, recall, F1 score, and false positive rate. Among the tested algorithms, XGBoost and decision trees stood out with high performance, offering superior accuracy and recall levels compared to other methods. KNN and SVM also showed good performance, although slightly lower than XGBoost and DT. Logistic regression and Naive Bayes, while less performant in some cases, demonstrated commendable robustness and execution speed.

These results highlight the importance of choosing the appropriate algorithm based on the specificities of the data and the requirements of the intrusion detection system.

Keywords: Artificial Intelligence, Intrusion Detection System (IDS), Cyberattacks, Machine Learning, CIC-IDS2017

ملخص

في العقدين الماضيين، أدى التطور السريع لتقنيات المعلومات والاتصالات، لا سيما الإنترنت، إلى موجة غير مسبوقة من الابتكارات على مستوى العالم. ومع ذلك، يصاحب هذا التقدم زيادة كبيرة في الهجمات الإلكترونية، مما يعرض أمن وخصوصية وتوفر الشبكات الحاسوبية للخطر. يمكن أن تتسبب الاختراقات في تعطيل العمليات بشكل خطير، وتسبب خسائر مالية كبيرة، وتلحق الضرر بسمعة المؤسسات. في مواجهة هذه التهديدات المتزايدة، يصبح حماية شبكات الاتصال تحديًا رئيسيًا للمستقبل. تعتبر أنظمة كشف التسلل (IDS) أساسية كخط دفاع أول، حيث كانت موضوعًا للبحث المكثف لقدرتها على تحديد ومنع الأنشطة الضارة قبل أن تتسبب في أضرار لا يمكن إصلاحها.

يقدم هذا العمل للبحث في مجالين واعدين من علوم الكمبيوتر: الذكاء الاصطناعي وكشف التسلل، بهدف تطوير نموذج قادر على الكشف التلقائي وتصنيف مجموعة واسعة من الهجمات. تشمل مساهمات هذا البحث في استكشاف طرق مختلفة للتعلم الآلي، ودراسة معمقة لتقنيات التعلم الآلي المختلفة، وتحليل النتائج لتقييم فعالية الأساليب المختلفة. تم تقييم أداء مختلف خوارزميات التعلم في سياق كشف التسلل باستخدام مجموعة البيانات العامة CIC-IDS2017. بعد معالجة البيانات وتطبيعها، تم تطبيق هذه الخوارزميات للتصنيف وتمت مقارنة أدائها باستخدام مقاييس تقييم مختلفة. الخوارزميات المستخدمة في هذه الدراسة تشمل الأشجار القرار (DT)، وNaive Bayes (NB)، وXGBoost، وآلات الدعم المتجه (SVM)، والجيران الأقرب (KNN)، والانحدار اللوجستي (LR). تظهر النتائج أن كل خوارزمية تقدم أداءً متنوعًا من حيث الدقة والاستدعاء وF1-score ومعدل الإيجابيات الكاذبة. من بين الخوارزميات المختبرة، تميزت XGBoost والأشجار القرار بأدائها العالي، حيث قدمت مستويات فائقة من الدقة والاستدعاء مقارنة بالطرق الأخرى. كما أظهرت KNN وSVM أداءً جيدًا، رغم أنهما أقل قليلاً من XGBoost وDT. الانحدار اللوجستي وNaive Bayes، على الرغم من أنهما أقل أداءً في بعض الحالات، أظهرا قوة وسرعة تنفيذ ملحوظة.

تؤكد هذه النتائج على أهمية اختيار الخوارزمية المناسبة بناءً على خصوصيات البيانات ومتطلبات نظام كشف التسلل.

الكلمات المفتاحية: الذكاء الاصطناعي، نظام كشف التسلل (IDS)، الهجمات الإلكترونية، التعلم الآلي، CIC-IDS2017.

Table de matière

INTRODUCTION GENERALE **2**
1. Contexte et Enjeux 2
2. Problématique etDéfis Actuels 2
3. Motivations et Perspectives 2
4. Organisationdu manuscrit 3
Chapitre1:Généralités sur IA, Machine Learning **4**
1. Introduction 5
2. Intelligence artificielle 5
3. Apprentissage automatique 6
3.1. Types d'apprentissage automatique 6
3.1.1. Apprentissage supervisé 6
3.1.2. Apprentissage non supervisé 7
3.1.3. Apprentissage semi-supervisé 7
3.1.4. Apprentissage par renforcement 7
3.1.5. Apprentissage par transfert Learning 7
3.2. Techniques d'apprentissage automatique 7
3.2.1. K-Nearest Neighbors 7
3.2.2. Arbre de décision 8
3.2.3. Forêt aléatoire 8
3.2.4. Naive Bayes 8
3.2.5. Xgboost 9
4. Conclusion 20
Chapitre 2:Les Systèmes de Détection d'Intrusion **21**
1. Introduction 22
2. Définition 22
3. Présentation d'un Système de Détection d'Intrusion 22
4. Les types des Systèmes de détection d'intrusions 29
4.1. Systèmes de détection d'intrusion en réseau 30
4.2. Systèmes de détection d'intrusion basée sur l'hôte 30
4.3. Le système hybride de détection d'intrusions 31
5. Les méthodes de détection du dépoilement IDS 32
5.1. détection basée sur les anomalies 32
5.2. détection basée sur les signatures 33
5.3. Comparaison entre les deux approches hybride 33
6. Les différents types d'attaques dans les IDSs 33
6.1. Un cheval de Troie 34
6.2 Un rootkit 34

6.3 Les Ransomware 34
6.4 keylogger

6.5 Les « grayware » 35
6.6.Un logiciel publicitaire 35
6.6 Le malvertising 35
6.7 Les logiciels espions 35
6.8 Un bot 36
6.7 Un logiciel malveillant hybride 36
7 .Les Avantages d'utilisation des IDS 37
7.2 Comprendre les Risques 37
7.3 Avertis ,Elaborer une stratégie d'une sécurité 38
7.4 Conformité réglementaire 38
7.5 Des temps des réponses plus rapides 38
8 État de l'art 38
9 Conclusion 40
Chapitre 3:Conception,Experimentationet Résultats 41
1. Introduction 42
2. Architecture de notre système 42
3. Présentation détaillée du système 43
3.1. Choix du Dataset 43
3.2. Prétraitement 53
3.2.1. Labellisation (étiquetage) 53
3.2.2. Nettoyage des données 54
3.2.3. Normalisation desdonnées 54
3.2.5.Étapededivision 55
3.3. Classification 55
3.3.1. Les algorithmes de l'apprentissage automatique 55
3.4. Les mesures d'évaluation des modèles 56
4. Résultat et discussion 56
4.1. Précision détaillée par classe 57
5. Conclusion 63
Conclusion générale 66
Bibliographie 68
Webographie 74
Annexe: Outils implémentation et application 71
1. Introduction .....
2. Matériel et logiciels utilisés .....
2.1. Python 72

2.2. Kagele 72

3. Bibliothèques Supplémentaires 76

3.1. NumPy.....

3.2. Pandas .....

3.3. Scikit-learn .....

3.4. TensorFlow..... 73

3.5. Keras 73

3.6. Matplotlib 73

3.7.Os 74

3.8.PyTorch 73

4. Étapes de classification..... 74

4.1. Importation des bibliothèques 74

4.2. Concaténation des tableaux 74

4.3. Étape de division 77

4.4. Création du modèle 78

4.5. Entrainement du modèle..... 79

5. Conclusion..... 80

Liste de figures

Chapitre1:Généralités sur Intelligence Artificielle ,Machine Learning

Figure1.1:Intelligence artificielle… 19

Figure1.2:Les méthodes d'apprentissage automatique… 21

Chapitre2:Les Systèmes de Détection d'Intrusions

Figure2.1:Processus de surveillance sur un SI.....29

Figure2.2:Endroit Systèmes de Détection d'Intrusions proposé par l'IDWG 31

Chapitre3:Conception, Expérimentation et Résultats

Figure3.1:Architecture globale de notre système.....42

Figure 3.2 : Nombre d'instances pour chaque Label 51

Figure3.3 : La matrice de confusion Knn n_voisins=3, metric='euclidean' ..... 60.

Figure3.4 La matrice de confusion Knn n_voisins=5, metric='euclidean' 62

Figure3.5. La matrice de confusion Knn n_voisins=3, metric='minkowski' 65

Figure3.6 : La matrice de confusion Knn n_voisins=5, metric='minkowski' 67

Figure3.7. La matrice de confusion Gaussian Naive Bayes..... 70

Figure3.8 : La matrice de confusion RF (metric=Entropy) 73

Figure3.9 : La matrice de confusion RF (metric=Gini).....74

Figure3.10 : La matrice de confusion XGBoost et Arbres de décision (Gini et Entropy).....78

Liste des tableaux

Chapitre2:Les Systèmes de Détection d'Intrusions

Tableau2.1:Les travaux connexes pour les systèmes de détection d'intrusion ..... 39..

Chapitre3:Conception, Expérimentation et Resultats

Tableau3.1:Description des fichiers contenant l'ensemble de données CIC-IDS2017 44

Tableau3.2 :Fonctionnalités de trafic réseau avec la description 56

Tableau3.3 :Numérisation des attaques 58

Liste des abréviations

ABIDS	Application-Base dIntrusion Detection **S**ystem
BNB	Bernoulli Naive Bayes
CNN	Réseaux des Neurones Convolutifs
DR	Dimensionality Reduction
DT	DecisionTree
GNB	GaussianNaiveBayes
HIDS	Host Base dIntrusions Detection System
IA	Intelligence Artificielle
IDS	Intrusions Detection System
IDWG	Intrusions Detection exchange format Working Group
KNN	K Nearest Neighbors
LR	Logistic Regression
ML	Machine Learning
MNB	Multinomial Naïve Bayes
MNIST	Base de Données de Chiffres Manuscrits
MSE	Mode Sans Échec
NB	Naive Bayes
NIDS	Network Based Intrusions Detection System
PCA	Principal Component Analysis
RF	Random Forest
RL	Reinforcement Learning
RNN	Réseaux des Neurons Récurrents
SL	**S**upervised Learning
SVM	Support Vector Machine
TL	Transfer Learning
UL	Unsupervised Learning

Introduction Générale

Introduction Générale

L'avènement de l'ère numérique, marqué par l'essor fulgurant d'Internet et des technologies de l'information au cours des deux dernières décennies, a profondément transformé notre société. Cette révolution a engendré une vague d'innovations sans précédent, redéfinissant nos modes de vie, nos pratiques commerciales et nos infrastructures essentielles. Cependant, cette mutation numérique s'accompagne d'une prolifération alarmante des cyber-menaces, mettant en péril l'intégrité, la confidentialité et la disponibilité des réseaux informatiques mondiaux.

1. Contexte et Enjeux

Face à l'escalade des cyber-attaques, la sécurisation des réseaux de communication s'impose comme un défi crucial pour notre avenir numérique. Les systèmes de détection d'intrusion (IDS) émergent comme un rempart indispensable contre ces menaces, constituant la première ligne de défense de nos infrastructures digitales. Leur rôle est d'examiner en permanence le trafic réseau, d'identifier les comportements suspects et de déclencher des alertes précoces, prévenant ainsi les dommages potentiellement irréversibles.

2. Problématique et Défis Actuels

Malgré les progrès significatifs réalisés dans le domaine des IDS, les approches conventionnelles de détection, souvent basées sur des analyses de seuils simplistes, peinent à suivre l'évolution rapide et la sophistication croissante des cyber-attaques modernes. Ces méthodes traditionnelles souffrent d'un manque de précision et de réactivité, exposant les organisations à des risques financiers conséquents et surchargeant les infrastructures réseau.

3. Motivations et Perspectives

L'ampleur et la complexité des cyber-menaces actuelles appellent à l'élaboration de solutions novatrices et performantes, capables d'assurer une détection rapide et précise des anomalies.Dans ce contexte, l'intelligence artificielle et l'apprentissage automatique se présentent comme des outils prometteurs pour révolutionner la sécurité des réseaux, offrant des capacités de détection des comportements anormaux nettement supérieures en termes de précision et d'efficacité.

4. Contribution et Approche Méthodologique

Ce travail se situe à l'intersection de deux domaines en pleine effervescence : l'intelligence artificielle et la détection d'intrusions. Notre objectif principal est de concevoir un modèlecapable de détecter et de classifier automatiquement un large spectre d'attaques informatiques. Pour y parvenir, nous avons mené une exploration approfondie de diverses méthodes d'apprentissage automatique.

Notre étude s'appuie sur l'analyse comparative de plusieurs algorithmes, incluant les arbres de décision (DT), Naive Bayes (NB), XGBoost, Les forêts aléatoires (RF) et les k-plus proches voisins (KNN). Ces méthodes ont été évaluées sur la base de données publique CIC-IDS2017, préalablement traitée et normalisée. Les performances de chaque algorithme ont été mesurées selon des critères rigoureux : précision, rappel, mesure F1 et taux de faux positifs.

Les résultats de notre analyse révèlent des performances variables selon les algorithmes, avec une distinction notable pour XGBoost et les arbres de décision, qui démontrent une efficacité supérieure dans la détection des intrusions.

Cette étude met en lumière l'importance cruciale d'une sélection judicieuse des algorithmes, adaptée aux spécificités des données traitées et aux exigences particulières des systèmes de détection d'intrusions. En intégrant les technologies de pointe de l'intelligence artificielle et de l'apprentissage automatique, nous ouvrons la voie à une amélioration significative de la sécurité des réseaux informatiques, contribuant ainsi à réduire l'impact et la fréquence des cyberattaques dans notre monde interconnecté.

5. Organisation du manuscrit

Ce manuscrit est organisé comme suit :

Chapitre 01 : Ce chapitre est consacré à présenter les concepts généraux relatifs l'intelligence artificielle (IA),machine Learning(ML)

Chapitre 02 : Ce chapitre est consacré à présenter une architecture globale d'un système de détection d'instrusion (IDS), la définition et le mode de fonctionnement de ce dernier. Ainsi la classification et les types des IDSs et la méthode de détection d'une intrusion. Enfin l'état de l'art sur les IDSs.

Chapitre03:Il regroupe la description de notre système, ainsi queles différentes expérimentations réalisées et l'analyse des résultats obtenus.

L’annexe contient un complément : Outils et environnement de réalisation et Bibliothèques essentielles en Python

Chapitre1:

Généralités sur Intelligence Artificielle ,

Machine Learning

1. Introduction

Dans ces dernières années, la société moderne a considérablement changé avec l'expansion rapide des technologies d'intelligence artificielle, d'apprentissage automatique qui impactent presque tous les aspects technologiques de l'entreprise. Le réseau neuronal est une technique d'apprentissage automatique inspirée par la fonctionnalité sophistiquée du cerveau humain et qui ressembleau système nerveux humain et à la structure du cerveau. Il peut être utilisé pour résoudre des problèmes.

Ce chapitre présente un groupe des techniques d'apprentissage automatique et des techniques basées sur la technologie du réseau neuronal.

2 .Intelligence Artificielle (IA)

2.1. Historique

l'intelligence artificielle débute en 1943, avec la publication de l'article « A Logical Calculus of Ideas Immanent in Nervous Activity. » par Warren McCullough et Walter Pitts qu'ils présentent le premier modèle mathématique pour la création d'un réseau de neurones.

En 1950, Alan Turing publie le Turing Test qui sert encore pour évaluer les IA. De ce test découlent les fondations de l'intelligence artificielle, de sa vision et de ses objectifs (répliquer ou simuler l'intelligence humaine dans les machines).

Mais ce n'est qu'en 1956, que le terme d'intelligence artificielle sera prononcé pour la première fois, durant la conférence « Dartmouth Summer Research Project on Artificial Intelligence. » de John McCarthy. Lors de cet événement, les chercheurs présentent les objectifs et la vision de l'IA. Beaucoup considèrent cette conférence comme la véritable naissance de l'intelligence artificielle telle qu'elle est connue. Plusieurs années s'écoulent et les travaux sur l'intelligence artificielle continuent. En 1959, Arthur Samuel invente le terme de Machine Learning en travaillant chez IBM. En 1989, le français Yann Lecun met au point le premier réseau de neurones capable de reconnaître des chiffres écrits à la main, cette invention sera à l'origine du développement du DeepLearning.[w1]

2.2 .Définition

Selon John McCarthy, le terme « intelligence artificielle » (Artificial Intelligence ou AI, en anglais) caractérise l'élaboration de programmes informatiques capables de prendre en charge destâches habituellement effectuées par des humains car demandant un apprentissage, une organisation de la mémoire et un raisonnement. L'objectif est de parvenir à transmettre à une machine des fonctions propres au vivant : rationalité, raisonnement, mémoire et perception.

Dans le domaine de l'IA, L'apprentissage automatique (en anglais Machine Learning, ML) se concentre sur le développement d'algorithmes permettant aux systèmes d'apprendre à partir des données sans une programmation explicite. À son tour, l'apprentissage profond (en anglais Deep Learning, DL) une sous- discipline du ML.[1]

(Voirfigure1.1)

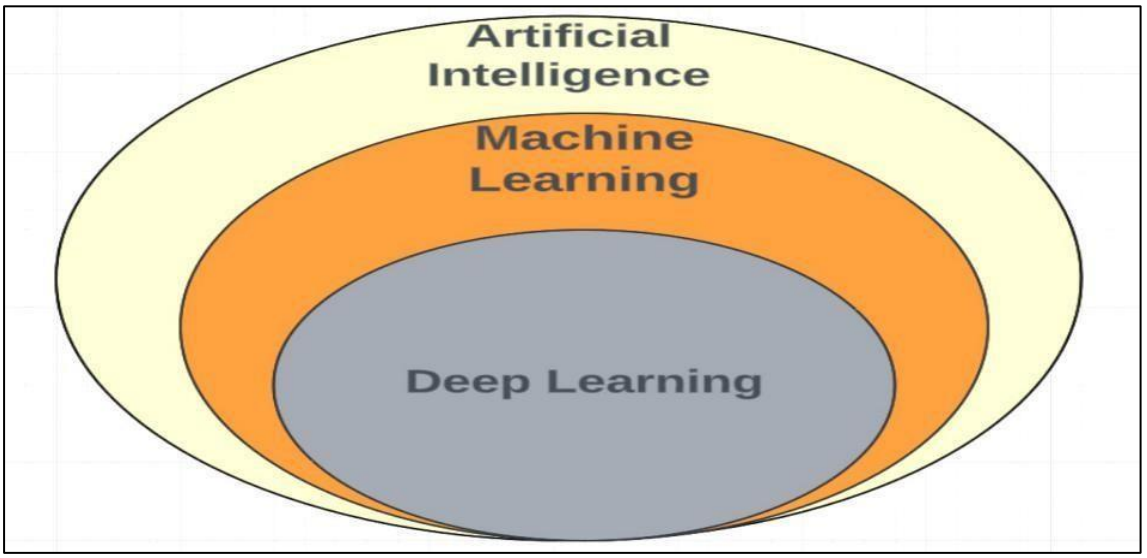

Figure1.1:Intelligenceartificielle

2.3 .Fonctionnement De l'IA

les systèmes d'IA fonctionnent en ingérant de grandes quantités de données d'apprentissage étiquetées, en analysant les données à la recherche de corrélations et de modèles, et en utilisant ces modèles pour faire des prédictions sur les états futurs. Cela nécessite une base d'infrastructure et de logiciels spécialisés pour l'écriture et la formation des algorithmes d'apprentissage automatique (Python, Java, C++) qui ont des caractéristiques très prisées par les développeurs d'IA. La programmation de l'IA se concentre sur les compétences cognitives L'apprentissage, Raisonnement, L'autocorrection, Créativité.[w2]

2.4 .Différences entre l'IA, l'apprentissage automatique et l'apprentissage profond

L'IA, l'apprentissage automatique et l'apprentissage profond sont des termes courants dans l'informatique d'entreprise et parfois utilisés de manière interchangeable, en particulier par les entreprises dans leurs documents marketing. Mais il existe des distinctions. Le terme "IA", inventé dans les années 1950, fait référence à la simulation de l'intelligence humaine par des machines. Il recouvre un ensemble de capacités en constante évolution au fur et à mesure que de nouvelles technologies sont développées. Les technologies qui relèvent de l'IA comprennent l'apprentissage automatique et l'apprentissage profond.

L'apprentissage automatique permet aux applications logicielles de devenir plus précises dans la prédiction des résultats sans être explicitement programmées pour le faire. Les algorithmes d'apprentissage automatique utilisent des données historiques pour prédire de nouvelles valeurs de sortie. Cette approche est devenue beaucoup plus efficace avec l'apparition de grands ensembles de données sur lesquels s'entraîner. L'apprentissage profond, un sous-ensemble de l'apprentissage automatique, est basé sur notre compréhension de la structure du cerveau. L'utilisation par l'apprentissage profond de la structure des réseaux neuronaux artificiels est à la base des récentes

avancées en matière d'IA, notamment les voitures auto-conduites et ChatGPT développé par OpenAI.[w2]

3 .Apprentissage automatique(Machine Learning (ML))

3.1. Définition

Le Machine Learning (ML) est une forme d'intelligence artificielle (IA) qui est axée sur la création de systèmes qui apprennent, ou améliorent leurs performances, en fonction des données qu'ils traitent. Le Machine Learning et l'IA sont souvent abordés ensemble et ces termes sont parfois utilisés de manière interchangeable bien qu'ils ne renvoient pas exactement au même concept. Une distinction importante est que, même si l'intégralité du Machine Learning repose sur l'intelligence artificielle, cette dernière ne se limite pas au Machine Learning.

Aujourd'hui, nous utilisons le Machine Learning dans tous les domaines. Lorsque nous interagissons avec les banques, achetons en ligne ou utilisons les médias sociaux, des algorithmes de Machine Learning entrent en jeu pour optimiser, fluidifier et sécuriser notre expérience. Le Machine Learning et la technologie qui l'entoure se développent rapidement, et nous commençons seulement à entrevoir ses capacités. [w3]

(Voir figure1.2)

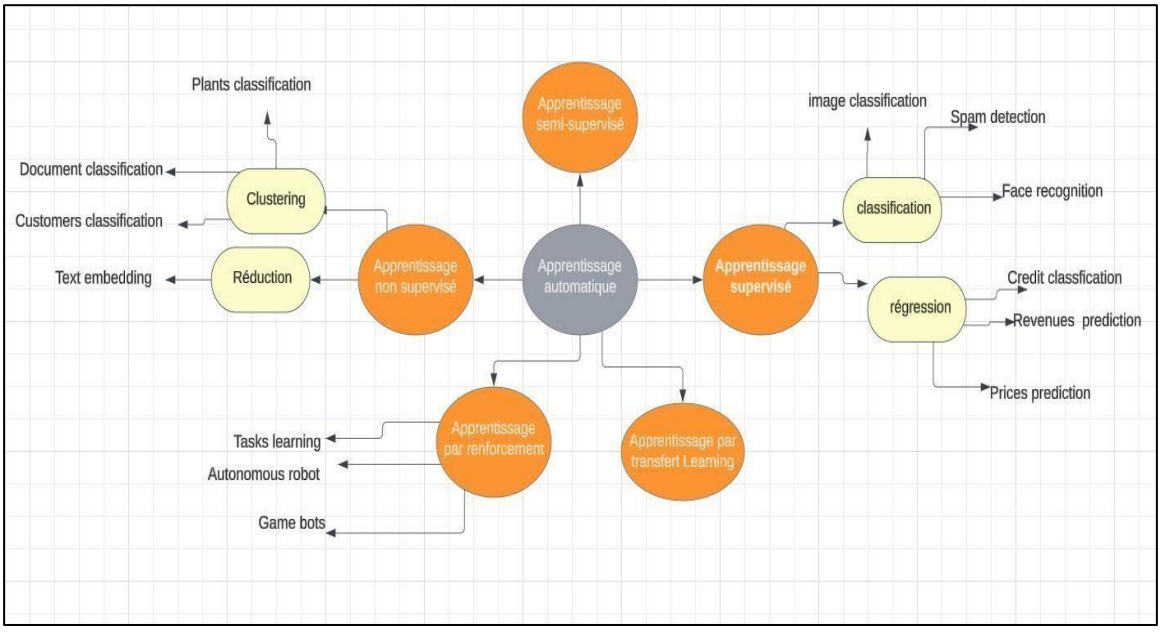

Figure1.2:Les méthodes d'apprentissage automatique

3.2. Types d'apprentissage automatique

Les algorithmes sont les moteurs du Machine Learning. En général, deux principaux types d'algorithmes de Machine Learning sont utilisés aujourd'hui : l'apprentissage supervisé et l'apprentissage non supervisé. La différence entre les deux se définit par la méthode employée pour traiter les données afin de faire des prédictions.[w4]

3.2.1 Machine Learning supervisé

Le Machine Learning supervisé est une technologie élémentaire mais stricte. Les opérateurs présentent à l'ordinateur des exemples d'entrées et les sorties souhaitées, et l'ordinateur recherche des solutions pour obtenir ces sorties en fonction de ces entrées. Le but est que l'ordinateur apprenne la règle générale qui mappe les entrées et les sorties.

Le machine Learning supervisé peut être utilisé pour faire des prédictions sur des données indisponibles ou futures (on parle alors de "modélisation prédictive"). L'algorithme essaie de développer une fonction qui prédit avec précision la sortie à partir des variables d'entrée – par exemple, prédire la valeur d'un bien immobilier (sortie) à partir d'entrées telles que le nombre de pièces, l'année de construction, la surface du terrain, l'emplacement, etc.

Le machine Learning supervisé peut se subdiviser en deux types :

- **Classification :** la variable de sortie est une catégorie.
- **Régression :** la variable de sortie est une valeur spécifique.

Les principaux algorithmes du Machine Learning supervisé sont les suivants : forêts aléatoires, arbres de décision, algorithme K-NN (k-Nearest Neighbors), régression linéaire, algorithme de Naïve Bayes, machine à vecteurs de support (SVM), régression logistique et boosting de gradient.

3.2.2 Machine Learning non supervisé

Dans le cadre du Machine Learning non supervisé, l'algorithme détermine lui-même la structure de l'entrée (aucune étiquette n'est appliquée à l'algorithme). Cette approche peut être un but en soi (qui permet de découvrir des structures enfouies dans les données) ou un moyen d'atteindre un certain but. Cette approche est également appelée « apprentissage des caractéristiques » (Feature Learning).

Un exemple de Machine Learning non supervisé est l'algorithme de reconnaissance faciale prédictive de Facebook, qui identifie les personnes sur les photos publiées par les utilisateurs.

Il existe deux types de Machine Learning non supervisé :

- **Clustering** : l'objectif consiste à trouver des regroupements dans les données.
- **Association** : l'objectif consiste à identifier les règles qui permettront de définir de grands groupes de données.

Les principaux algorithmes du Machine Learning non supervisé sont les suivants : K-Means, Clustering/regroupement hiérarchique et réduction de la dimensionnalité

3.2.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé est une approche qui combine à la fois des éléments supervisés et non supervisés, où seule une partie des données est étiquetée. Cette partie étiquetée est ensuite utilisée pour déduire des informations sur la partie non étiquetée, comme c'est le cas dans les systèmes de récupération de texte ou d'image.

3.2.4 Apprentissage par renforcement

L'apprentissage par renforcement (en anglais reinforcement learning, RL) se distinguedeSL et UL en enseignant à une machine à exécuter une séquence d'actions. Leschercheurs programment un algorithme pour accomplir une tâche dans le RL, fournissant des rétroactions positives ou négatives pendant qu'il travaille sur la façon d'exécuter la tâche. Le programmeur définit les règles pour les récompenses, mais laisse à l'algorithme laliberté de déterminer

Les étapes à suivre pour maximiser la récompense et accomplir la tâche .

3.2.5 Apprentissage par transfert Learning

L'apprentissage par transfert(en anglais transfer learning,TL)nous commençons par former unréseau de base sur un ensemble de données et une tâche de base, puis nous réutilisons ou transférons les fonctionnalités apprises vers un second réseau cible, qui sera entraîné sur un ensemble de données et une tâche cibles distincts. Ce processus réussit généralement lorsque les fonctionnalités sont générales, adaptées à la fois aux tâches de base et cibles, plutôt que spécifiques à la tâche de base [w1].

3.3 .Techniques d'apprentissage automatique

Parmi les nombreuses techniques d'apprentissage automatique, certaines se distinguent par leur efficacité et leur utilisation répandue:

3.3.1 K-Nearest Neighbors

Les K plus proches voisins (en anglais K-Nearest Neighbors, KNN) une méthode de Machine Learning supervisée, se révèle être une approche simple mais efficace pour la classification et la régression. Cette méthode repose sur la similarité entre les données, mesurée à l'aide de diverses techniques telles que la distance euclidienne, deChebyshev ou encore cosinus. Ce qui distingue KNN, c'est sa rapidité, car il ne nécessite pasde phase de formation préalable pour générer des prédictions. Cependant, sa performance diminue lorsque la taille des données augmente, ce qui constitue son principal inconvénient. De plus, les données de grande dimension peuvent compliquer les calculs[2].

3.3.2 Arbre de décision

L'arbre de décision (en anglais DecisionTree, DT) une méthode de Machine Learning supervisée, est utilisé pour résoudre des problèmes de classification et de régression.

Cet algorithme crée un modèle à partir des données d'entraînement, utilisant des règles de décision simples pour prédire la classe ou la valeur d'une variable cible. La prédiction commence par le nœud racine de l'arbre et progresse jusqu'au nœud terminal. Contrairement à d'autres méthodes, l'arbre de décision nécessite moins de préparation des données lors

du prétraitement, et la normalisation des données n'est pas requise. Cependant, de petites variations dans les données peuvent entraîner des changements significatifs dans la structure de l'arbre, ce qui peut rendre le modèle instable. De plus, l'entraînement d'un arbre de décision peut être chronophage et parfois le calcul devient très complexe[3].

3.3.3 Forêt aléatoire

La forêt aléatoire (en anglais Random Forest, RF) est une méthode flexible et largement utilisée en apprentissage supervisé, adaptée à la fois à la classification et à la régression[4]. Pour assurer des estimations stables et précises, RF construit plusieurs arbres de décision à l'aide du bagging, puis les combine. Ses paramètres par défaut sont souvent efficaces pour produire des prédictions précises tout en restant facilement interprétables. Cependant, RF peut être lent et inefficace dans des scénarios nécessitant une classification en temps réel avec un grand nombre d'arbres.

3.3.4 Naive Bayes

L'algorithme Naïve Bayes (NB), basé sur le théorème de Bayes, est prisé pour sa simplicité de construction et son efficacité, surtout avec de vastes ensembles de données[5].Contrairement aux variables numériques, il s'adapte bien aux variables catégorielles. Il offre diverses variantes comme le Naïve Bayes gaussien (GNB), multinomial (MNB), de Bernoulli (BNB), entre autres. Un défi majeur réside dans sa tendance à attribuerune probabilité nulle aux variables catégorie les absentes dans l'ensemble d'entraînement, rendantla prédiction impossible pour ces variables dans l'ensemble de test. Toute fois,des techniques de lissage peuvent être employées pour pallier ce problème de fréquence zéro. Malgré ses avantages, il est important de souligner que le Naïve Bayes est généralement considéré comme une stimateur peu fiable, constituant ainsi une limite[6].

3.3.5 Xgboost

Xgboost (eXtreme Gradient Boosting) est une technique de Machine Learning très populaire chez les Data Scientists. Il s'agit d'un modèle amélioré de l'algorithme d'amplification de gradient (Gradient Boost). Cette technique d'apprentissage machine est également utilisé pour résoudre les problématiques courantes d'entreprises tout en se basant sur une quantité minimale de ressources.

En Machine Learning ,le boosting de gradient extrême est une méthode qui est utilisée

pour réduire le nombre d'erreurs dans l'analyse prédictive des données. XGBoost est un assemblage d'arbres décisionnels (weaklearners) qui prédisent les résidus et corrige les erreurs des arbres décisionnels précédents. La particularité de cet algorithme réside dans l'arbre décisionnel utilisé[w5].

4.Conclusion

Nous avons consacré ce chapitre à la présentation des notions de base telles que l'IA, les concepts fondamentaux de l'apprentissage automatique. Nous avons d'abord défini l'ML et présenté ses différents types afin de fournir une vision globale de ce domaine. En suite, nous avons expliqué les algorithmes, et certains algorithmes clés.

Dans le prochain chapitre, nous présenterons le système de détection d'intrusion

Chapitre2:

Les Systèmes de Détection D ’Intrusions

1. Introduction

Les Systèmes de Détection d'Intrusions (en anglais Intrusion Détection System, IDS) sont des systèmes logiciels ou matériels qui automatisent le processus de surveillance des événements se produisant dans un système informatique fectuant une certaine analyse pour détecter les signes de problèmes de sécurité. Les IDSs sont devenus nécessaires pour assurer la sécurité des organisations. En effet, les IDSs sont considérés comme un système défense efficaces contre les attaques réseau dirigées contre les systèmes informatiques.

Dans ce chapitre nous présentons les concepts d'IDS ainsi que son architecture. Nous prenons plusieurs critère set Nous commençons par la classification selon la méthode d'analyse qui découpe les IDS en deux approches (comportementale et par signatures), enfin nous allons mettre le point sur la détection d'intrusions réseau.

2. Définition

En sécurité informatique, un système de détection d'intrusion (ou IDS : Intrusion Détection System) est un mécanisme destiné à repérer des activités anormales ou suspectes sur la cible analysée (un réseau ou un hôte). Il permet ainsi d'avoir une connaissance sur les tentatives réussies comme échouées des intrusions.

Il existe deux grandes catégories d'IDS, les plus connues sont les détections par signatures (reconnaissance de programme malveillant) et les détections par anomalies (détecter les écarts par rapport à un modèle représentant les bons comportements, cela est souvent associé a de l'apprentissage automatique). Il est aussi possible de classifier les IDS selon la cible qu'ils vont surveiller, les plus communs sont les systèmes de détection d'intrusion réseau et les systèmes de détection d'intrusion hôte.

Certains IDS ont la possibilité de répondre aux menaces qu'ils ont détectées, ces IDS avec capacité de réponse sont des systèmes de prévention d'intrusion.[w6]

Figure2.1:Processus de surveillance sur un SI

3. Présentation d'un Systèmes de Détection d'Intrusions

l'architecture IDWG d'un système de détection d'intrusion contient des capteurs qui collectent les données brutes vues comme une suite d'octets (reflet direct d'une activité) depuis une source de données (interface réseau, journaux système, etc.), détecte les événements et les envoient à un analyseur. Un ou plusieurs capteurs associés à un analyseur constituent l'environnement d'une sonde de détection d'intrusion. L'analyseur est un processus chargé de filtrer et de qualifier les

événements. Il lève une alerte si un événement est intéressant (*Event of Interest* ou EOI) au regard de la sécurité du système d'information. Ce peut-être l'ouverture inattendue d'une session Telnet, les entrées des fichiers de journaux système qui retracent un utilisateur qui tente d'accéder à des fichiers dont il n'a pas les droits d'accès ou des échecs de connexion répétés, etc. Les alertes (et donc les événements associés) sont transmises de l'analyseur vers le gestionnaire d'alertes (manager) qui notifie un opérateur humain (console de supervision, trappe SNMP, envoi d'un e-mail). Les événements doivent être convertis dans un format de message d'alertes (e.g. IDMEF) avant d'être transmis. Cette normalisation est nécessaire pour homogénéiser les alertes indépendamment des modèles de sondes et des sources de données. L'opérateur de sécurité valide les alertes et met en place des contre-mesures appropriées (réponses) qui peuvent être automatisées. L'administrateur sécurité (composante humaine) configure la sonde et le gestionnaire d'alertes conformément à la politique de sécurité. Il est à noter que l'opérateur et l'administrateur peuvent être la même personne.

Dans le cadre d'un environnement réel, il n'y a pas d'architecture type. Certains IDS vont combiner ces entités fonctionnelles dans un seul module. D'autres solutions vont s'appuyer sur desapplications externes pour compléter le système de détection d'intrusion .

Tous les équipements de sécurité (pare-feu, systèmes de détection d'intrusion, etc.) ou les systèmes et équipements réseau (routeurs, serveurs, etc.) d'un système d'information susceptibles de remonter des événements, vus comme des signaux, permettent de construire une perception de l'état du système d'information et donc participent à la détection d'intrusion. Tous ces événementsqui se présentent sous la forme de données brutes doivent pouvoir faire l'objet d'un pré-filtrage à lasource avant d'être triés et qualifiés par le moteur de corrélation. Ainsi, certains événements inscrits dans les journaux du système d'exploitation et des applications peuvent être écartés de l'analyse.

Une architecture globale de supervision peut également être construite autour d'une solution très structurante comme le SOC (*Security Operation Center* ou centre opérationnel de sécurité). Le SOC permet non seulement de couvrir un large spectre de données provenant de différentes sources (modules IDS/IPS, pare-feu, routeur, postes de travail, application, serveur, équipement réseau, etc.), mais aussi d'avoir une vision globale de la sécurité du système supervisé.

L'architecture SOC est composée d'une base de connaissance de toutes les informations relatives à l'infrastructure réseaux et systèmes (e.g. les adresses IP, les noms des machines, les applications installées sur chaque machine et leur version, etc.) et des vulnérabilités (systèmes d'exploitation,

applications, réseaux, etc.) qui pourraient être exploitées par un attaquant pour s'introduire dans le système. Ces informations structurelles permettent de coupler une information d'état telle que la vulnérabilité d'un serveur à une information d'action telle que la détection de cette attaque à destination du serveur en question.

Avant de configurer les sondes, de concevoir des règles d'analyse et de corrélation des alertes, il est nécessaire d'évaluer à chaque changement du système à superviser et à chaque nouvelle vulnérabilité découverte, le niveau de sécurité d'une l'infrastructure IT.

Cette évaluation permet de déterminer si une intrusion sur le système est possible et si elle peut s'avérer critique. L'évaluation de la criticité d'un événement est une des opérations les plus difficiles dans la mesure où elle prend en compte, d'une part, l'importance de la ressource cible et,d'autre part, l'impact de l'attaque sur cette ressource. Les tests d'intrusion font partie intégrante decette évaluation et participent à améliorer la qualité du diagnostic.

Le SOC va également tenir compte des aspects de la politique de sécurité en termes de contrôle d'accès (e.g. l'authentification des administrateurs) et d'opérations autorisées (e.g. le scan de ports depuis une adresse IP spécifique dans le cadre d'un audit de sécurité). Tout autre comportement traité par les sondes sera alors vu comme une tentative d'intrusion.[w7]

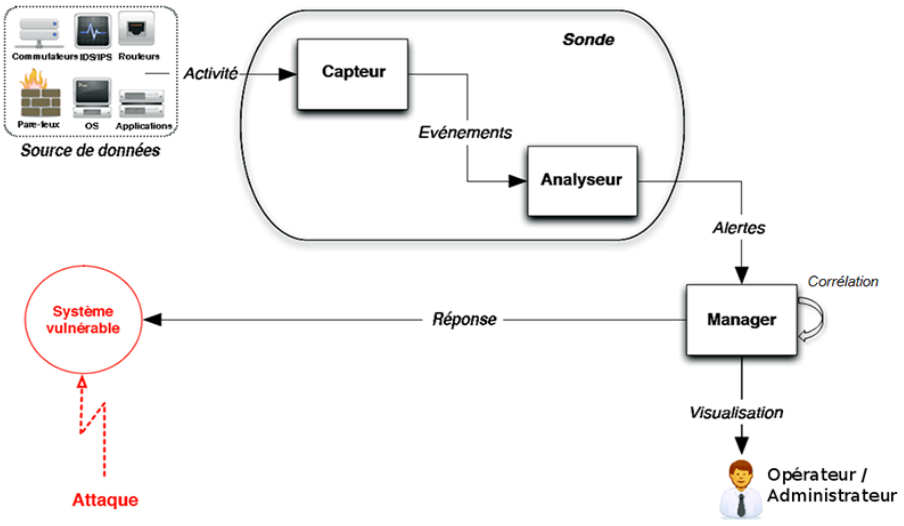

Figure2.2: Systèmes de Détection d'Intrusions proposé par l'IDWG

4.Les types des systèmes de détection des intrusions

Les systèmes de détection des intrusions sont conçus pour être déployés dans différents environnements. Et comme de nombreuses solutions de cybersécurité, un IDS peut être basé sur l'hôte ou sur le réseau.[w8]

4.1 **.IDS basé sur l'hôte (HIDS) :** un IDS basé sur l'hôte est déployé sur un poste particulier et conçu pour le protéger contre les menaces internes et externes. Un tel IDS peut avoir la capacité de surveiller le trafic réseau en provenance et à destination de la machine, d'observer les processus en cours et d'inspecter les journaux du système. La visibilité d'un IDS basé sur l'hôte est limitée à la machine hôte, ce qui réduit le contexte disponible pour la prise de décision, mais il dispose d'une visibilité approfondie sur les parties internes de l'ordinateur hôte.

4.2.IDS basé sur le réseau (NIDS) : une solution IDS basée sur le réseau est conçue pour surveiller l'ensemble d'un réseau protégé. Il a une visibilité sur l'ensemble du trafic circulant sur le réseau et prend des décisions basées sur les métadonnées et le contenu des paquets. Ce point de vue élargi permet de mieux comprendre le contexte et de détecter desmenaces généralisées ; toutefois, ces systèmes manquent de visibilité sur les éléments internes du poste qu'ils protègent.

En raison des différents niveaux de visibilité, le déploiement d'un HIDS ou d'un NIDS de manière isolée n'offre qu'une protection incomplète au système d'une organisation. Une solution de gestion unifiée des menaces, qui intègre plusieurs technologies en un seul système, peut offrir une sécurité plus complète.

5. Méthode de détection du déploiement IDS

Au-delà de leur emplacement de déploiement, les solutions IDS diffèrent également dans la manière dont elles identifient les intrusions potentielles :[w9]

5.1 .**Détection par signature** : Les solutions IDS basées sur des signatures utilisent les empreintes digitales des menaces connues pour les identifier. Une fois que le logiciel malveillant ou un autre contenu malveillant a été identifié, une signature est générée et

ajoutée à la liste utilisée par la solution IDS pour tester le contenu entrant. Cela permet à un IDS d'atteindre un taux élevé de détection des menaces sans faux positifs, car toutes les alertes sont générées sur la base de la détection d'un contenu malveillant connu. Cependant, un IDS basé sur des signatures est limité à la détection des menaces connues et ne tient pas compte de la vulnérabilité des "zero-day".

5.2.Détection d'anomalies : Les solutions IDS basées sur les anomalies construisent un modèle du comportement "normal" du système protégé. Tous les comportements futurs sont comparés à ce modèle et toute anomalie est qualifiée de menace potentielle et génère des alertes. Si cette approche permet de détecter les nouvelles menaces ou les menaces de type "zero-day", la difficulté de construire un modèle précis de comportement "normal" signifie que ces systèmes doivent équilibrer les faux positifs (alertes erronées) et les faux négatifs (détections manquées).

5.3..Détection hybride : Un IDS hybride utilise à la fois la détection basée sur les signatures etla détection basée sur les anomalies. Cela lui permet de détecter davantage d'attaques potentielles avec un taux d'erreur plus faible qu'en utilisant l'un ou l'autre système de manière isolée.

6. Les différents types d'attaques dans les IDSs

Il ya plusieurs attaques : [w10]

6.1 .**Un cheval de Troie** : ou trojan est un logiciel malveillant qui trompe les utilisateurs sur sa véritable intention en se faisant passer pour un programme légitime. Le terme est dérivé de l'histoire grecque ancienne du cheval de Troie trompeur qui a entraîné la chute de la ville de Troie. Les chevaux de Troie sont généralement diffusés par le biais d'une ingénierie sociale telle que le phishing. Par exemple, un utilisateur peut être incité à exécuter une pièce jointe à un courrieldéguisé pour paraître authentique (par exemple, une feuille de calcul Excel). Une fois le fichier exécutable ouvert, le cheval de Troie est installé.

6.2 **Un rootkit** : est un ensemble de logiciels malveillants conçus pour donner un accès non autorisé à un ordinateur ou à une partie de son logiciel et qui masquent souvent leur existence ou celle d'autres logiciels. L'installation d'un rootkit peut être automatisée ou le pirate peut l'installeravec un accès administrateur. L'accès peut être obtenu à la suite d'une attaque directe sur le système, comme l'exploitation de vulnérabilités, le craquage de mots de passe ou l'hameçonnage. La détection d'un rootkit est difficile car il peut subvertir le programme antivirus

destiné à le détecter. Les méthodes de détection comprennent l'utilisation de systèmes d'exploitation fiables, les méthodes comportementales, l'analyse des signatures, l'analyse des différences et l'analyse du contenu de la mémoire.

6.3. **Les Ransomware** : sont des logiciels malveillants conçus pour empêcher l'accès à un système informatique ou à des données jusqu'à ce qu'une rançon soit versée. Les rançongiciels se propagent par le biais de courriels d'hameçonnage, de publicités malveillantes, de la visite de sitesweb infectés ou de l'exploitation de vulnérabilités. Les attaques de ransomware entraînent des temps d'arrêt, des fuites de données, des vols de propriété intellectuelle et des violations de données. Le montant des rançons varie de quelques centaines à des centaines de milliers de dollars. Payables en crypto-monnaies comme le bitcoin.

6.4 keylogger :qui Les enregistreurs de frappe, les enregistreurs de frappe ou la surveillance du système sont un type de logiciel malveillant utilisé pour surveiller et enregistrer chaque frappe effectuée sur le clavier d'un ordinateur spécifique. Des enregistreurs de frappe sont également disponibles pour les smartphones. Les enregistreurs de frappe stockent les informations recueillies et les envoient au pirate qui peut alors extraire des informations sensibles telles que les identifiants de connexion et les détails des cartes de crédit.

6.5 Les « grayware » : a été inventé en septembre 2004 et décrit des applications ou des fichiers indésirables qui ne sont pas des logiciels malveillants mais qui détériorent les performances de l'ordinateur et peuvent présenter un risque pour la cybersécurité. Au minimum, les graywares se comportent de manière gênante ou indésirable et, au pire, ils surveillent le système et transmettent des informations par téléphone. Les graywares font référence aux adwares et aux spywares. La bonne nouvelle, c'est que la plupart des logiciels antivirus peuvent détecter les programmes potentiellement indésirables et proposer de les supprimer.

6.6 Un logiciel malveillant sans fichier : Les logiciels malveillants sans fichier sont des logiciels malveillants qui utilisent des programmes légitimes pour infecter un ordinateur. Contrairement à d'autres infections, ils ne s'appuient pas sur des fichiers et ne laissent pas d'empreinte, ce qui complique la détection et la suppression par les logiciels anti-programmes malveillants. Il existe exclusivement en tant qu'artefact basé sur la mémoire de l'ordinateur, c'est-à-dire dans la RAM. Les logiciels malveillants sans fichier sont apparus en 2017 comme une cybermenace courante, mais ils existent depuis longtemps. Frodo, Number of the Beast et DarkAvenger sont lespremières attaques de logiciels malveillants sans fichier. Plus récemment, le Comité national

démocrate et la violation d'Equifax ont été victimes d'attaques de logiciels malveillants sans fichier.

6.7 Un logiciel publicitaire : est un type de logiciel gris conçu pour afficher des publicités sur votre écran, souvent dans un navigateur web ou une fenêtre contextuelle. Généralement, ils se présentent comme légitimes ou se greffent sur un autre programme pour vous inciter à les installer sur votre ordinateur, votre tablette ou votre smartphone. Les logiciels publicitaires sont l'une des formes de logiciels malveillants les plus rentables et les moins nocives, et ils sont de plus en plus populaires sur les appareils mobiles. Les logiciels publicitaires génèrent des revenus en affichant automatiquement des publicités à l'intention de l'utilisateur du logiciel.

6.8 Le malvertising : qui signifie « publicité malveillante », est l'utilisation de la publicité pour diffuser des logiciels malveillants. Il s'agit généralement d'injecter des publicités malveillantes ou chargées de logiciels malveillants dans des réseaux publicitaires et des pages web légitimes.La publicité est un excellent moyen de diffuser des logiciels malveillants, car des efforts considérables sont déployés pour rendre les annonces attrayantes pour les utilisateurs, afin de vendre ou de faire la publicité d'un produit. Le malvertising bénéficie également de la réputation des sites sur lesquels il est placé, tels que les sites d'information réputés.

6.9 Les logiciels espions : sont des logiciels malveillants qui recueillent des informations sur une personne ou une organisation, parfois à son insu, et les envoient à l'attaquant sans le consentement de la victime. Les logiciels espions ont généralement pour but de suivre et de vendre vos données d'utilisation de l'internet, de capturer les informations de votre carte de crédit ou de votre compte bancaire ou de voler des informations personnelles identifiables (PII).Certains types de logiciels espions peuvent installer des logiciels supplémentaires et modifier les paramètres de votre appareil. Les logiciels espions sont généralement faciles à supprimer car ils ne sont pas aussi dangereux que d'autres types de logiciels malveillants.

6.10Un bot : est un ordinateur infecté par un logiciel malveillant qui lui permet d'être contrôlé à distance par un pirate. Le bot (ou ordinateur zombie) peut ensuite être utilisé pour lancer d'autres cyberattaques ou faire partie d'un botnet (ensemble de bots).Les botnets sont une méthode populaire pour les attaques par déni de service distribué (DDoS), la diffusion de logiciels rançonneurs, l'enregistrement des frappes et la diffusion d'autres types de logiciels malveillants.

6.11.Une porte dérobée : est une méthode secrète permettant de contourner l'authentification ou le cryptage normal d'un ordinateur, d'un produit, d'un dispositif intégré (par exemple, un routeur) ou d'une autre partie d'un ordinateur. Les portes dérobées sont généralement utilisées pour sécuriser l'accès à distance à un ordinateur ou pour accéder à des fichiers cryptés. À partir de là, elles peuvent être utilisées pour accéder à des données sensibles, les corrompre, les supprimer ou les transférer. Les portes dérobées peuvent prendre la forme d'une partie cachée d'un programme (un cheval de Troie), d'un programme séparé ou d'un code dans les microprogrammes et les systèmes d'exploitation.

6.12Un pirate de navigateur : ou hijackware modifie le comportement d'un navigateur web en envoyant l'utilisateur vers une nouvelle page, en changeant sa page d'accueil, en installant des barres d'outils indésirables, en affichant des publicités non désirées ou en dirigeant les utilisateurs vers un site web différent.

6.13.Les logiciels malveillants : sont une catégorie de logiciels malveillants conçus pour automatiser la cybercriminalité. Il est conçu pour perpétrer des vols d'identité par le biais de l'ingénierie sociale ou de la furtivité afin d'accéder aux comptes financiers et de vente au détail de la victime pour voler des fonds ou effectuer des transactions non autorisées. Il peut également voler des informations confidentielles ou sensibles dans le cadre de l'espionnage d'entreprise.

6.14 Une application mobile malveillante :Toutes les applications disponibles sur l'App Store et Google Play ne sont pas légitimes. Cela dit, l'App Store est généralement plus sûr en raison d'une meilleure présélection des applications tierces. Les applications malveillantes peuvent voler des informations sur les utilisateurs, tenter de les extorquer, accéder aux réseaux d'entreprise, les forcer à voir des publicités indésirables ou installer une porte dérobée sur l'appareil.

6.15Un RAM scraper : est un type de logiciel malveillant qui exploite les données temporairement stockées en mémoire ou en RAM. Ce type de logiciel malveillant cible souvent les systèmes de point de vente (POS) tels que les caisses enregistreuses, car ils peuvent stocker des numéros de carte de crédit non cryptés pendant une brève période avant de les crypter et de les transmettre au back-end.

6.16Les logiciels de sécurité malveillants : font croire à l'utilisateur que son système présente unproblème de sécurité, tel qu'un virus, et l'incitent à payer pour le faire disparaître. En réalité, le faux logiciel de sécurité est le logiciel malveillant qui doit être supprimé.

6.17 Le crypto jacking : est un type de logiciel malveillant qui utilise la puissance de calcul de lavictime pour extraire de la crypto-monnaie.

6.18 un logiciel malveillant hybride :la plupart des logiciels malveillants sont une combinaison d'attaques de logiciels malveillants existants, souvent des chevaux de Troie, des vers, des virus et des logiciels rançonneurs.

6.19L'ingénierie sociale et le phishing : ne sont pas des logiciels malveillants à proprement parler. Il s'agit de mécanismes de diffusion populaires pour les attaques de logiciels malveillants. Par exemple, un hameçonneur peut essayer d'amener un utilisateur à se connecter à un site web d'hameçonnage, mais il peut également joindre une pièce jointe infectée à l'e-mail pour augmenter ses chances de réussite.

6.20 Un bug :Tout comme l'ingénierie sociale et le phishing, les bugs ne sont pas des logiciels malveillants, mais ils peuvent ouvrir des vulnérabilités que les logiciels malveillants peuvent exploiter. Un bon exemple est la vulnérabilité EternalBlue qui se trouvait dans les systèmes d'exploitation Windows et qui a conduit à la propagation du crypto-logiciel de rançon WannaCry.

7. Avantages des systèmes de détection d'intrusion

Les solutions IDS offrent des avantages majeurs aux organisations, principalement en identifiant les menaces de sécurité potentielles pour leurs réseaux et leurs utilisateurs. Voici quelques avantages courants du déploiement d'un IDS :[w11]

7.1 **Comprendre les risques** : un outil IDS aide les entreprises à comprendre le nombre d'attaques qui leur sont ciblées, ainsi que le type et le niveau de sophistication des risques auxquels elles sont confrontées.

7.2. **Élaborer une stratégie de sécurité** : la compréhension des risques est cruciale pour établir etfaire évoluer une stratégie de cybersécurité complète qui peut résister au paysage des menaces moderne. Un IDS peut également être utilisé pour identifier les bugs et les défauts potentiels des appareils et des réseaux des organisations, puis évaluer et adapter leurs défenses pour traiter les risques auxquels elles peuvent être confrontées à l'avenir.

7.3. **Conformité réglementaire** : les organisations sont désormais confrontées à une liste en constante évolution de réglementations de plus en plus strictes auxquelles elles doivent se

conformer. Un outil IDS leur offre une visibilité sur ce qui se passe sur leurs réseaux, ce qui facilite le processus de conformité à ces réglementations. Les informations qu'il collecte et enregistre dans ses journaux sont également essentielles pour que les entreprises documentent qu'elles répondent à leurs exigences de conformité.

7.4. **Des temps de réponse plus rapides** : les alertes immédiates que les solutions IDS initient permettent aux organisations de détecter et de prévenir les attaquants plus rapidement qu'elles ne le feraient grâce à une surveillance manuelle de leurs réseaux. Les capteurs qu'un IDS utilise peuvent également inspecter les données des paquets réseau et des systèmes d'exploitation, ce qui est également plus rapide que la collecte manuelle de ces informations.

8. État de l'art

L'application de l'apprentissage automatique a pris une importance croissante pour aborder certaines détections d'intrusion émergentes. Grâce à leur capacité à traiter directement les données brutes, les méthodes d'apprentissage automatique permettent d'apprendre des caractéristiques et de réaliser des classifications simultanément. Cela les rend l'approche de référence dans les études sur les systèmes de détection d'intrusion (voir Tableau).Parallèlement, leshackers publient leurs performances et les nouvelles méthodes qu'ils ont créées, tandis que lessystèmes de détection d'intrusion sont continuellement développés pour être de plus en plusefficaces contre ces malwares.

Etudes	**Année**	**Méthode**	**Jeux dedonnées utilisés**	**Mesures deperformance**	**Valeurs**
Salama et al.[37]	2011	DBN-SVM	NSL-KDD	Accuracy	90%

Ferraget al.[38]	2020	models. (DNN, RNN,CNN)Unsupervisedmodels (RBM, DBN,DBM,DA)	CSE-CICIDS2018	Accuracy	97.37%
Patil et al.[39]	2019	Random Forest.	CSE-CIC-IDS2017 UNSW-NB15	Accuracy	99.96% 96.94%
Pujol-Perichet al.[40]	2020	GNN (graph neuralnetwork)	CSE-CICIDS2018	Accuracy	99%
Kanimozhiet al.[41]	2019	ANN,RF, k-NN, SVM,Adaboost,NB	CSE-CICIDS2018	Accuracy Precision RecallScore-F1	99.91% 99.95% 99.98% 99.93%
Alomet al.[43]	2015	DBN	NSL-KDD	Accuracy	97.5%
Wuetal.[44]	2018	CNN	NSL-KDD	Accuracy	97.88%

Tableau2.2:Les travaux connexes pour les systèmes de détection d'intrusion

9. Conclusion

Dans ce chapitre, nous avons présenté le system détection d'intrusions(IDS). nous avons d'abord Donné une définition à cette notion, puis nous avons présenté une classification des systèmes de détection d'intrusions (IDS).Cette classification comporte les IDS baséshôte (HIDS, les IDS basés sur l'application (AB-IDS)les IDS basés réseau(N- IDS).Nous avons abordé aussi les deux grandes approches utilisées par les Modes de détection IDS àsavoir: La détection d'anomalies et La reconnaissance de signature , puis déroule sur les différents attaques sur le système (IDS)

Dans le chapitre suivant, nous présenterons la structure générale de notre système de détection d'intrusions , ainsi que le protocole expérimental suivi et les résultats obtenus.

Chapitre 3:

Conception, Expérimentation et Résultats

1. Introduction

Ces dernières années, avec la généralisation d'Internet, le nombre croissant d'ordinateurs inter connectés dans notre quotidien a été remarquable. Cependant, cette expansion a également exposé les vulnérabilités des serveurs, facilitant l'accès des pirates informatiques à ces systèmes par le biais non seulement de méthodes d'attaque bien connues, mais aussi de nouvelles approches émergentes.

Dans ce chapitre, nous nous concentrerons sur la conception et la mise en œuvre de systèmes de détection d'intrusion (IDS) exploitant les techniques de ML et de DL.

Nous détaillerons les étapes clés de notre recherche, en commençant par la présentation de la base de données utilisée et les prétraitements nécessaires pour l'adapter à notre étude. Ensuite, nous aborderons le choix des algorithmes sélectionnés, puis nous passerons à l'apprentissage des modèles où nous diviserons les données en ensembles d'apprentissage et detest. Enfin, nous procéderons à une analyse approfondie des différentes mesures de performance qui seront utilisées pour comparer ces algorithmes.

2.Architecture de notre système

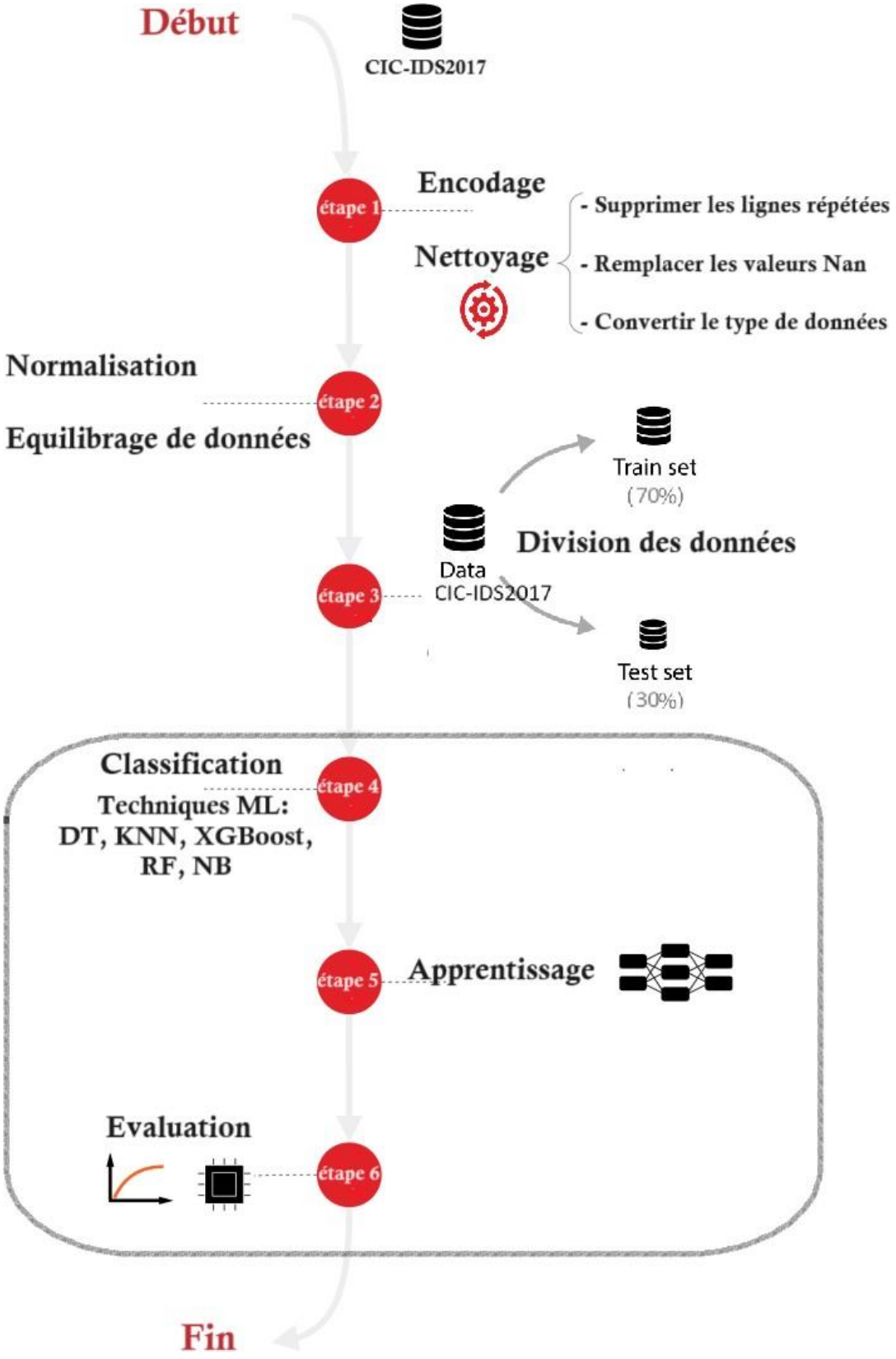

Figure 3.1:Architecture globale de notre système

2. Présentation détaillée du système

2.1. Choix du Dataset

Depuis sa création, l'ensemble de données CIC-IDS2017(Canadian Institute for Cybersecurity Intrusion Détection Dataset 2017) a suscité l'intérêt des chercheurs pour l'analyse et le développement de nouveaux modèles et algorithmes. Ce dataset se compose de 8 fichiers distincts contenant des données de trafic normal et d'attaques sur une périodede5 jours[w7]. CIC-IDS2017 inclut à la fois des attaques simulées et des exemples d'attaques courantes, reproduisant ainsi des scénarios similaires aux données réelles du monde.L'ensemble de donnée s'intègre également les résultats de l'analyse du trafic réseau à l'aide de CICFlowMeter. Les flux sont étiquetés en fonction de l'horodatage, des adresses IP source et destination, des ports source et destination, des protocoles, ainsi que des types d'attaques.CICFlowMeter génère des flux bidirectionnels (Biflow), où chaque flux est caractérisé parplus de 80 fonctionnalités, incluant des statistiques telles que la durée ,le nombre de paquets, le nombre d'octets, la longueur des paquets, etc.Ces caractéristiques sont calculées séparément pour les directions avant(source vers destination)et arrière (destination vers source).

Les données sont exportées au format CSV avec 6 colonnes principales: FlowID, SourceIP, DestinationIP, SourcePort, DestinationPort, et Protocol.

En résumé, le tableau 3.2 présente une vue détaillée de chaque fonctionnalité de trafic réseau avec sa description respective, offrant ainsi une base solide pour l'analyse et le développement de techniques avancées de détection d'intrusion[45].

Nom des fichiers	**Activité de Jour**	**Attaques trouvées**
MondayWorkingHours.pcap_ISCX.csv	Monday	Benign(normal activities)

TuesdayWorkingHours.pcap_ISCX.csv	Tuesday	Benign,FTP-Patator, SSH-Patator
WednesdayworkingHours.pcap_ISCX.csv	Wednesday	Benign,DoSGoldenEye,DoSHulkDoSSlowhttptest, DoSslowloris,Heartbleed
ThursdayWorkingHoursMorningWebAttacks. pcap_ISCX.csv	Thursday	Benign,WebAttack–BruteForce, WebAttak–SqlInjection, WebAttackXSS
Thursday-WorkingHours-AfternoonInfilteration.pcap_ISCX.csv	Thursday	Benign,Infiltration
Friday-WorkingHoursMorning.pcap_ISCX.csv	Friday	Benign,Bot
Friday-WorkingHourAfternoonPortScan.pcap_ISCX.csv	Friday	Benign,PortScan
Friday-WorkingHours-AfternoonDDos.pcap_ISCX.csv	Friday	Benign,DDoS

Tableau 3.1 : Description des fichiers contenant l'ensemble de données CIC-IDS2017

Le tableau 3.2 montre que l'ensemble de données contient des informations sur les attaques sous la forme de données de trafic sur 5 jours. Les données du jeudi après-midi et du vendredi conviennent bien à la classification binaire. De même, les données du mardi, du mercredi et du jeudi matin conviennent mieux à la conception d'un modèle de détection multi classe. Il convient toutefois de noter qu'un modèle de détection optimal doit être capable de détecter les attaques de tout type. Par conséquent, pour concevoir un tel IDS typique, les données de trafic de tous les jours doivent être fusionnées pour former un seul ensemble de données à utiliser par l'IDS. C'est exactement ce que nous avons fait pour fusionner ces fichiers. En fusionnant les fichiers présentés dans le tableau 3.2, nous avons trouvé la forme complète d'un ensemble de données qui contient 3119345 instanceset 78 caractéristiques contenant 15 étiquettes de classe (1 normale + 14 étiquettes d'attaque). En outre, l'examen des instances des fichiers combinés a révélé que l'ensemble de données contient 288602 instances dont l'étiquette de classe est manquante et 203 instances dont l'information est manquante. En supprimant ces instances manquantes, nous avons obtenu un ensemble de données combiné deCIC-IDS2017 contenant 2830540

instances. À ce stade, nous avons recherché d'éventuelles instances redondantes. De manière sur prenante, aucune instance redondante n'a été trouvée. Les caractéristiques de l'ensemble de données combiné et l'occurrence détaillée par classe sont présentées dans le tableau 3.3 et le tableau3.4 [w8].

Remarque:

Nom de la fonction: les noms des fonctionnalités (caractéristiques) présentes dans le jeu de données CIC-IDS2017.

Numéro des classes:le nombre de classes uniques présentes dans chaque colonne

Nomdelafonction	**Description**	**Numérodesclasses**
FlowID	un identifiant unique attribué auxfluxderéseau.	1085071
SourceIP	L'adresse IP de source de laconnexion.	17005
SourcePort	Portdesource dela connexion.	64640
DestinationIP	L'adresse IP de destination de laconnexion.	19112
DestinationPort	Portdedestinationdelaconnexion.	53805
Protocol	Protocoleutilisélorsdelaconnexion	3
Timestamp	Tempsàlaquellelaconnexionaeulieu	27965
FlowDuration	Duréedufluxenmicrosecondes	1050899
TotalFwd Packets	Totaldespaquets danslesensdirect	1432
TotalBackwardPackets	Total des paquets dans le sensinverse	1747
TotalLengthofFwdPackets	Taille totale des paquets dans lesensdirect	17928
Total Length of BwdPackets	Taille totale du paquet dans le sensinverse	64698
FwdPacket LengthMax	Taille maximale du paquet dans ladirectionavant	5279
Fwd PacketLength Min	Taille minimale du paquet dans ladirectionavant	384

FwdPacket LengthMean	Taille moyenne du paquet dans ladirectionavant	109091
FwdPacket LengthStd	taille du paquet dans la directionavant	254384
BwdPacket LengthMax	Taillemaximaledupaquetensens inverse	4838
Bwd PacketLengthMin	Tailleminimaledupaquetdansle	583
	sensinverse	
BwdPacket LengthMean	Taillemoyenne dupaquetensens inverse	154284
BwdPacket LengthStd	tailledupaquetdanslesensinverse	249206
FlowBytes/s	Nombred'octetsdefluxparse conde	1595244
FlowPackets/s	Nombre de paquets de flux parseconde	1242273
FlowIAT Mean	Tempsmoyenentredeuxpaquetsen voyésdans leflux	1170377
FlowIATStd	Écart-typedutempsentredeuxpaquet senvoyés dansleflux	1057046
FlowIATMax	Temps maximum entre deuxpaquetsenvoyésdanslefl ux	580289
FlowIATMin	Temps minimum entre deuxpaquetsenvoyésdanslefl ux	136316
FwdIAT Total	Tempstotalentredeuxpaquetsen voyésdanslesensdirect	493098
FwdIATMean	Tempsmoyenentredeuxpaquetsen voyésdanslesensdirect	738963
Fwd IATStd	Écart-type entre deux paquetsenvoyésdanslesensdi rect	700372
FwdIATMax	Temps maximum entre deuxpaquetsenvoyésdanslesensdire ct	437316
Fwd IAT Min	Temps minimum entre deuxpaquetsenvoyésdanslesensdire ct	110631

BwdIAT Total	Tempstotalentredeuxpaquetsen voyésensensInverse	414928
BwdIAT Mean	Tempsmoyenentredeuxpaquetsen voyésensensinverse	671985
BwdIAT Std	Écart-typedutempsentredeuxpaquetsenvoyés dans le sensinverse	709055
BwdIAT Max	Tempsmaximumentredeux	368285
	paquetsenvoyés dans le sensinverse	
Bwd IATMin	Temps minimum entre deuxpaquets envoyésdanslesensinverse	66074
FwdPSHFlags	Nombre de foisoùl'indicateur PSHa étéactivé dans les paquetscirculantdans lesens direct (0pour UDP	2
Bwd PSHFlags	Nombre de foisoùl'indicateur PSHa étéactivé dans les paquetsvoyageantdans lesens inverse(0pourUDP)	1
FwdURGFlags	Nombre de foisoùl'indicateur URGa étéactivé dans despaquetscirculantdans lesens direct (0pourUDP)	2
BwdURGFlags	Nombre de foisoùl'indicateur URGa étéactivé dans les paquetsvoyageant dans le sens inverse (0pourUDP)	1
FwdHeaderLength	Total des octets utilisés pour les en-têtesdanslesens direct	3771
BwdHeaderLength	Nombre total d'octetsutilisés pourlesen-têtesdanslesensinverse	3945
FwdPackets/s	Nombre de paquetsaller parseconde	1222680
BwdPackets/s	Nombre de paquetsensens inverseparseconde	1109941
MinPacketLength	Longueurminimaled'unpaquet	215
MaxPacketLength	Longueurmaximaled'unpaquet	5708

PacketLength Mean	Longueurmoyenned'unpaquet	226511
PacketLength Std	Longueurdel'écart-typed'unpaquet	413401
PacketLengthVariance	Variancedela longueurd'unpaquet	408238
FINFlagCount	NombredepaquetsavecFIN	2
SYNFlagCount	NombredepaquetsavecSYN	2
RSTFlagCount	Nombredepaquetsavec RST	2
PSHFlagCount	NombredepaquetsavecPUSH	2
ACKFlagCount	Nombrede paquetsavecACK	2
URGFlagCount	Nombrede paquetsavecURG	2
CWEFlagCount	Nombredepaquetsavec CWR	2
ECEFlagCount	NombredepaquetsavecECE	2
Down/Patio	Taux de téléchargement et detéléversement	31
AveragePacket Size	Taillemoyennedespaquets	212207
AvgFwd Segment Size	Taille moyenneobservée dans lesensdirect	99719
AvgBwdSegmentSize	Taillemoyenneobservéedansle sensinverse	147611
FwdHeaderLength.1	Nombre total d'octetsutilisés pourlesen-têtesdansle sensdirect	3771
FwdAvgBytes /Bulk	Nombremoyend'octetsenvracdansl esensdirect	1
FwdAvg Packets/Bulk	Nombremoyendepaquetsenvracdansl esens direct	1
FwdAvg BulkRate	Nombremoyendepaquetsenvracdansl esens direct	1

BwdAvgBytes /Bulk	Nombremoyend'octetsenvracdans lesens inverse	1
BwdAvgPackets/Bulk	Nombremoyendepaquetsenvracdans lesens inverse	1
Bwd AvgBulkRate	Nombremoyend'octetsen massedanslesens inverse	1
SubflowFwd Packets	Nombremoyen de paquets dans unsous-fluxdanslesensdirect	1432
SubflowFwdBytes	Nombremoyend'octetsdans un	17928
	sous-fluxdanslesens direct	
SubflowBwd Packets	Nombremoyen de paquets dans unsous-fluxdanslesens inverse	1747
SubflowBwd Bytes	Le nombremoyend'octets dans unsous-fluxdansle sensinverse.	64738
Init_Win_bytes_forward	Nombre total d'octetsenvoyés danslafenêtreinitialedanslesensdirect	12151
Init_Win_bytes_backward	Nombretotald'octetsenvoyésdansla fenêtreinitiale dans le sensinverse.	13112
act_data_pkt_fwd	Nombre depaquetscontenantaumoins1octetdedonnées TCPdanslesens direct	1093
min_seg_size_forward	Taille minimale d'un segmentobservéedanslesens direct	28
ActiveMean	Durée moyenned'activité d'un fluxavantqu'ilnedevienneinactif	326583
Active Std	Écart-typedutempsd'activitéd'unfluxavantl'inactivité	202829
ActiveMax	Duréemaximaled'activitéd'unfluxavantqu'ilnedevienneinactif	299565
ActiveMin	Durée minimale pendant laquelle unflux a étéactifavant de devenirinactif	175670

IdleMean	Durée moyenned'inactivité d'unfluxavantqu'ilnedevienneact if	222137
Idle Std	Écart-type de la durée d'inactivitéd'un flux avantqu'il ne devienneactif	197617
IdleMax	Duréemaximaled'inactivitéd'unfl uxavantqu'ilnedevienneactif	149737
IdleMin	Durée minimaled'inactivité d'unfluxavantqu'ilnedevienneact if	223888
Label	représentelaclassificationdutraficrés eauen tant que normal oulié àune intrusion ou à uneattaqueparticulière	15

Tableau3.2: Fonctionnalités de trafic réseau avec la description

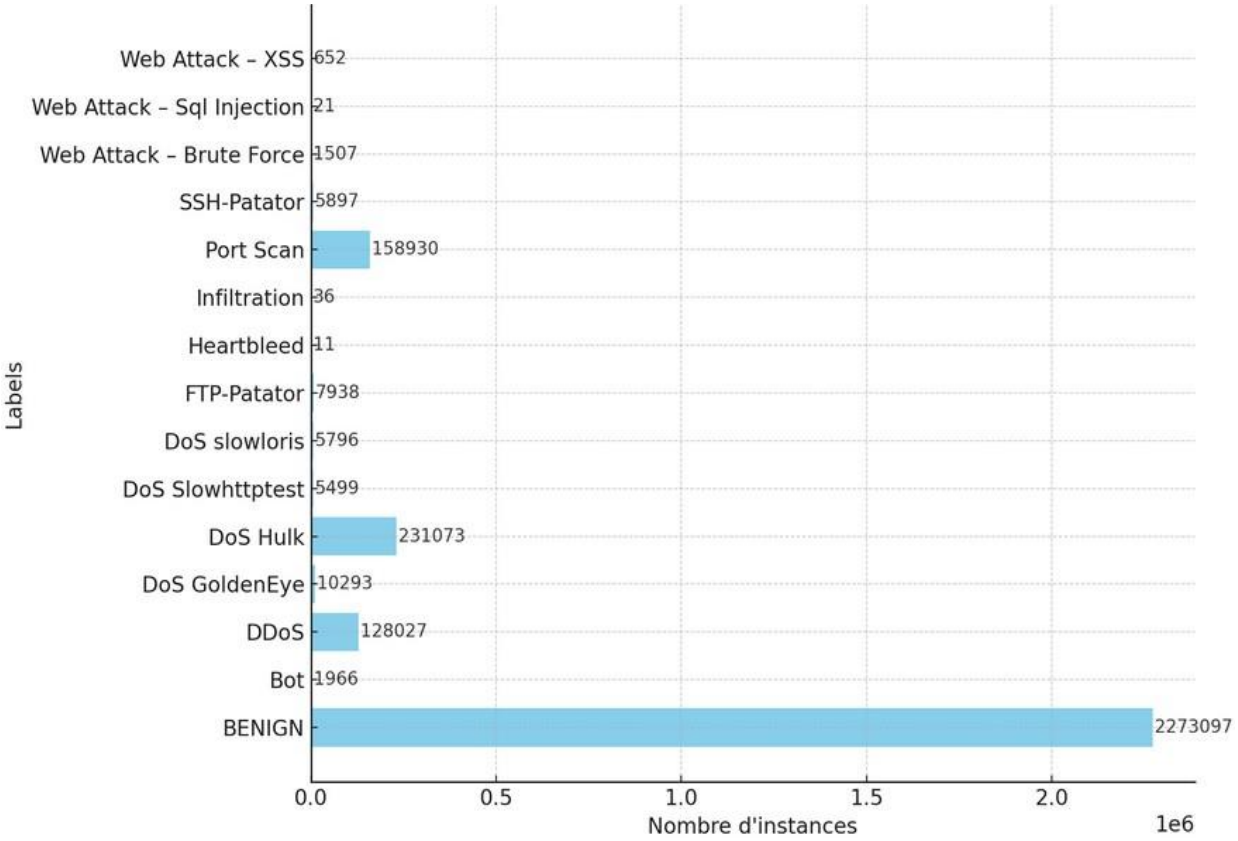
Web Attack – XSS
Web Attack – Sql Injection
Web Attack – Brute Force
SSH-Patator
Port Scan
Infiltration
Heartbleed
FTP-Patator
DoS slowloris
DoS Slowhttptest
DoS Hulk
DoS GoldenEye
DDoS
Bot
BENIGN
652
21
1507
5897
158930
36
11
7938
5796
5499
231073
10293
128027
1966
2273097
Labels
0.0
0.5
1.0
1.5
2.0
Nombre d'instances
1e6

Figure3.2:Nombre d’instances pour chaque Label

La figure (3.2) illustre la répartition du nombre d'occurrences pour différents types d'attaques ainsi que les instances bénignes (normales) dans un jeu de données de détection d'intrusion. Voici quelques observations clés :

1. Prévalence des Données Bénignes : La catégorie "BENIGN" (bénin) domine largement le graphe avec 2 273 097 instances. Cela suggère que les données normales sont beaucoup plus fréquentes que les données représentant des attaques, ce qui est typique dans les jeux de données de sécurité informatique.
2. Attaques DDoS et DoS : Parmi les attaques, les types les plus représentés sont "DoS Hulk" (231 073 instances) et "DDoS" (128 027 instances). Cela indique une prévalence relativement élevée de ces types d'attaques dans le jeu de données, reflétant peut-être leur fréquence dans des scénarios réels.
3. Port Scan et Bot : "Port Scan" (158 930 instances) et "Bot" (1966 instances) sont également notables. Le port scanning est une technique courante utilisée par les attaquants pour identifier les points faibles dans les systèmes, expliquant ainsi sa fréquence élevée.
4. Attaques Moins Fréquentes : Certaines attaques, telles que "Heartbleed" (11 instances), "Infiltration" (36 instances), et "Web Attack – Sql Injection" (21 instances) sont beaucoup moins fréquentes. Cela peut être dû à leur nature plus ciblée ou à leur rareté dans le scénario du jeu de données.
5. Autres Types d'Attaques : Les autres attaques, comme "DoSGoldenEye", "DoSSlowhttptest", "DoSslowloris", "FTP-Patator", "SSH-Patator", "Web Attack – Brute Force", et "Web Attack – XSS" montrent également des occurrences variables, reflétant une diversité des types d'attaques présentes dans le jeu de données.

Implications :

- Déséquilibre des Données : La grande disproportion entre les données bénignes et les attaques indique un déséquilibre qui peut poser des défis pour les modèles de machine learning, notamment en ce qui concerne la détection des attaques rares.

- Focus sur Attaques Fréquentes : Les algorithmes de détection d'intrusion devront être particulièrement efficaces pour détecter les types d'attaques plus fréquentes (comme DoS et DDoS) tout en restant sensibles aux attaques rares mais potentiellement dévastatrices.

2.2. Prétraitement

Le prétraitement des données est une technique d'exploration des données qui transformeles données brutes en formats compréhensibles, en s'attaquant à des problèmes tels que l'incomplétude, l'incohérence et les erreurs dans les grands ensembles de données[46].

3.2. La bellisation (étiquetage)

consiste à attribuer des étiquettes ou des catégories à des données pour les classifier. Cela permet aux modèles d'apprentissage automatique d'associer les caractéristiques des données à leurs étiquettes correspondantes, facilitant ainsi la prédiction de nouvelles données.

Label textuelle	Label numérique
BENIGN	0
Bot	1
DDoS	2
DoSGoldenEye	3
DoSHulk	4
DoSSlowhttptest	5
DoSslowloris	6
FTP-Patator	7
Heartbleed	8
Infiltration	9
PortScan	10
SSH-Patator	11
WebAttack–BruteForce	12

WebAttack–SqlInjection	13
WebAttack–XSS	14

Tableau3.3:Numérisation des attaques

3.2.2. Nettoyage des données

Assurer un nettoyage efficace des données est une étape cruciale avant l'analyse ou la modélisation des données, même si cette tâche peut s'avérer la borieuse. Nous avons commencé par supprimer les lignes .En double entant qu'étape initiale du processus de nettoyage des données. En outre, nous avons supprimé les espaces et remplacé les valeurs manquantes (NaN) par les moyennes descolonnes et convertir les colonnes "Object" en valeurs numériques. [w9]

3.2.3. Normalisation des données

La plupart du temps, en machine Learning, les Data Set proviennent avec des ordresde grandeurs différents. Cette différence d'échelle peut conduire à des performancesmoindres. Pour palier à cela, des traitements préparatoires sur les données existent. Notamment le caractéristique Scaling qui comprend la Standardisation et la Normalisation. Min- MaxScaling peut être appliqué quand les données varient dans des échelles différentes.A l'issue de cette transformation, les caractéristiques seront comprises dans un intervalle fixe[0,1]. Le but d'avoir un tel intervalle restreint est de réduire l'espace de variation des valeursd'une caractéristique et par conséquent réduire l'effet de valeurs aberrantes. La normalisationpeut être effectuée par la technique du Min-Max Scaling. La transformation se fait grâce à la formule suivante:

$$nrae= \frac{X- Xmin}{Xmax-Xmin}$$

Avec:

- Xmin: la plus petite valeur observée pour la caractéristique X
- Xmax: la plus grande valeur observée pour la caractéristique X
- X:La valeur de la caractéristique qu'on cherche à normaliser [w10]

3.2.4. Ré-équilibrage

En complément du choix d'un critère pertinent, il peut être intéressant de tenter deré-équilibrer l'échantillon pour aider les algorithmes à mieux détecter les individus de la classe minoritaire. Les méthodes classiques consistent à créer de nouvelles observations de la classe minoritaire(oversampling)et/ou supprimer des individus de la classe minoritaire(un de rsampl ing). et SMOTE pour Synthetic Minority Over Sampling Technique consiste à sur-échantillonner en se basant sur les proches voisins de la classe minoritaire.[47]

3.2.5. Étape de division

Dans cette étape, nous divisons l'ensemble de données en 70 % de données d'entraînement et 30% de données de Test.

3.3. Classification

Las élection de l'algorithme d'apprentissage est une étape critique dans notre étude, nécessitant une évaluation minutieuse des performances sur notre jeu de données. Nous avons exploré plusieurs approches pour identifier celle qui répond le mieux à nos objectifs de classification.

3.3.1. Les algorithmes de l'apprentissage automatique

Nous avons évalué les performances des principaux algorithmes d'apprentissage machine suivants:

- K-Nearest Neighbors
- Arbre de décision
- Forêt aléatoire
- Gaussian Naive Bayes
- X gboost

Chacun de ces algorithmes a été testé et comparé pour déterminer celui qui convient le mieux à

notre cas d'étude.

3.4. Les mesures d'évaluation des modèles

On a Les mesures suivants:

- Précision(Pr):le pourcentage des attaques identifiées comme des attaquesTP par mitous les exemples prédit comme attaque, cette métrique, également relative à chaque catégorie, renseigne sur la probabilité qu'une prédiction d'une catégorie donnée soit correcte.Il est donné par:

$$Pr=TP/(TP+FP)$$

- Accuracy (le taux de réussite) : Indique la façon dont la technique de détection est correcte. C'est une métrique qui traduit également le rapport entre les détections correctes et les détections totales obtenues.

$$Accuracy=(TP+TN)/(TP+TN+FP+FN)$$

- Recall (Rc) Le taux de détection(Rappel)ou bien TPR(True PositiveRate):le pourcentage des attaques identifiées comme des attaques TP parmi tous les attaques dans l'ensemble de données, C'est le rapport entre le nombre d'intrusions correctement détectées et le nombre total d'intrusions. Et décrit par la formule :

$$TPRoubienRc=TP/(TP+FN)$$

- F1-score (F1) : La moyenne harmonique F combine le rappel et la précision en un nombrecomprisentre0 et1, il est donné par:

$$F1=2*(Pr*Rc)/(Pr+Rc)$$

Expérimentation

Les expérimentations, que nous allons mener, constituent une étude comparative approfondie de différents algorithmes de machine learning appliqués à la détection d'intrusions réseau. Voici une description détaillée de ces expérimentations :

1. Objectif : Évaluer et comparer l'efficacité de plusieurs algorithmes de classification pour la détection d'intrusions réseau, en utilisant un ensemble de données commun.
2. Ensemble de données : Utilisation probable du dataset CIC-IDS2017, un ensemble de données public largement utilisé pour la recherche en détection d'intrusions.
3. Prétraitement des données :
    - Nettoyage et normalisation des données
    - Encodage des variables catégorielles
    - Division des données en ensembles d'entraînement et de test
4. Algorithmes évalués :
    - K-Nearest Neighbors (KNN) avec différentes configurations :
        - 3 et 5 voisins
        - Distances euclidienne et de Minkowski
    - GaussianNaiveBayes
    - Random Forest avec critères Gini et Entropy
    - XGBoost
    - Arbres de décision avec critères Gini et Entropy
5. Métriques d'évaluation :
    - Précision (Precision)
    - Rappel (Recall)
    - Score F1 (F1-score)
    - Accuracy globale
    - Matrices de confusion (implicites dans les rapports)
6. Procédure expérimentale :
    - Entraînement de chaque modèle sur l'ensemble d'entraînement
    - Prédiction sur l'ensemble de test
    - Génération de rapports de classification détaillés pour chaque modèle

KNN

1- K NeighborsClassifier(n_neighbors=3, metric='euclidean')

```
Confusion Matrix:
```

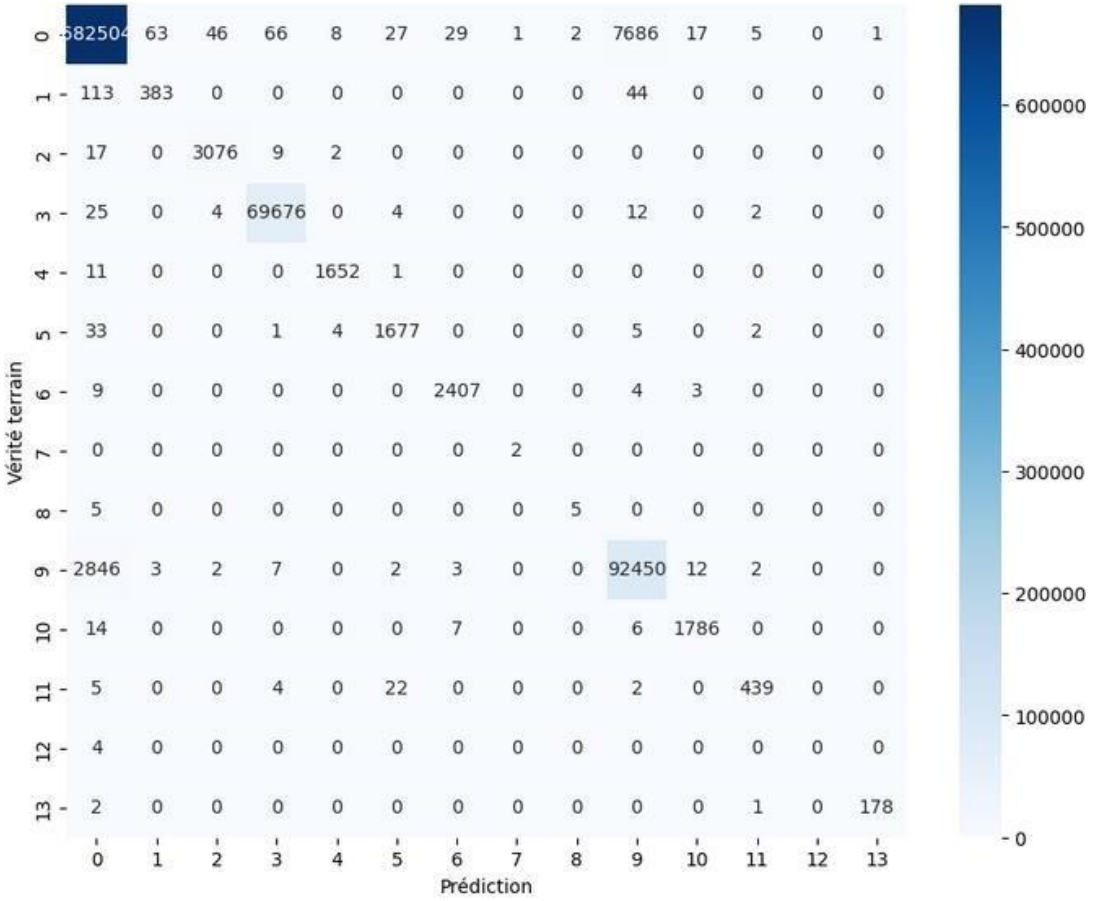

Figure3.3 : La matrice de confusion Knn n_voisins=3, metric='euclidean'

Rapport de classification KNN 3 euclidean:

	precision	recall	f1-score	support
0	0.9955	0.9885	0.9920	690455
1	0.8530	0.7093	0.7745	540
2	0.9834	0.9910	0.9872	3104
3	0.9988	0.9993	0.9990	69723
4	0.9916	0.9928	0.9922	1664
5	0.9677	0.9739	0.9708	1722
6	0.9841	0.9934	0.9887	2423
7	0.6667	1.0000	0.8000	2
8	0.7143	0.5000	0.5882	10
9	0.9226	0.9698	0.9456	95327
10	0.9824	0.9851	0.9838	1813
11	0.9734	0.9301	0.9512	472

12	0.0000	0.0000	0.0000	4
13	0.9944	0.9834	0.9889	181
accuracy			0.9871	867440
macro avg	0.8591	0.8583	0.8544	867440
weighted avg	0.9875	0.9871	0.9872	867440

Voici une analyse des résultats de classification du modèle KNN avec 3 voisins et la distance euclidienne :

1. Performance globale :
    - L'accuracy globale est de 98.71%, ce qui est excellent.
    - La moyenne pondérée des métriques (precision, recall, f1-score) est d'environ 98.7%, indiquant une très bonne performance générale.
2. Analyse par classe : a) Classes bien détectées (f1-score > 0.95) :
    - BENIGN (0), DDoS (2), DoS GoldenEye (3), DoS Hulk (4), DoS Slowhttptest (5), DoS slowloris (6), Port Scan (10), Web Attack – Sql Injection (13)
    - Ces classes sont très bien détectées avec un équilibre entre precision et recall.

    b) Classes moyennement détectées (0.8 < f1-score < 0.95) :

    - Bot (1), Infiltration (9), SSH-Patator (11)
    - Ces classes ont une bonne détection mais avec une marge d'amélioration.

    c) Classes mal détectées (f1-score < 0.8) :

    - FTP-Patator (7), Heartbleed (8), Web Attack – Brute Force (12), Web Attack – XSS (14)
    - Ces classes présentent des problèmes de détection.
3. Points spécifiques :
    - FTP-Patator (7) : Recall parfait mais precision faible, probablement dû au faible nombre d'échantillons (2).
    - Heartbleed (8) : Faible performance globale, possiblement due au petit nombre d'échantillons (10).

- Web Attack – Brute Force (12) : Échec total de détection (0 pour toutes les métriques).
- Web Attack – XSS (14) : Aucune métrique rapportée, suggérant un problème potentiel avec cette classe.

4. Déséquilibre des classes :
   - Certaines classes comme BENIGN (0) et Infiltration (9) ont beaucoup plus d'échantillons que d'autres.
   - Ce déséquilibre peut affecter la performance du modèle sur les classes minoritaires.

2- K NeighborsClassifier(n_neighbors=5, metric='euclidean')

```
Confusion Matrix:
```

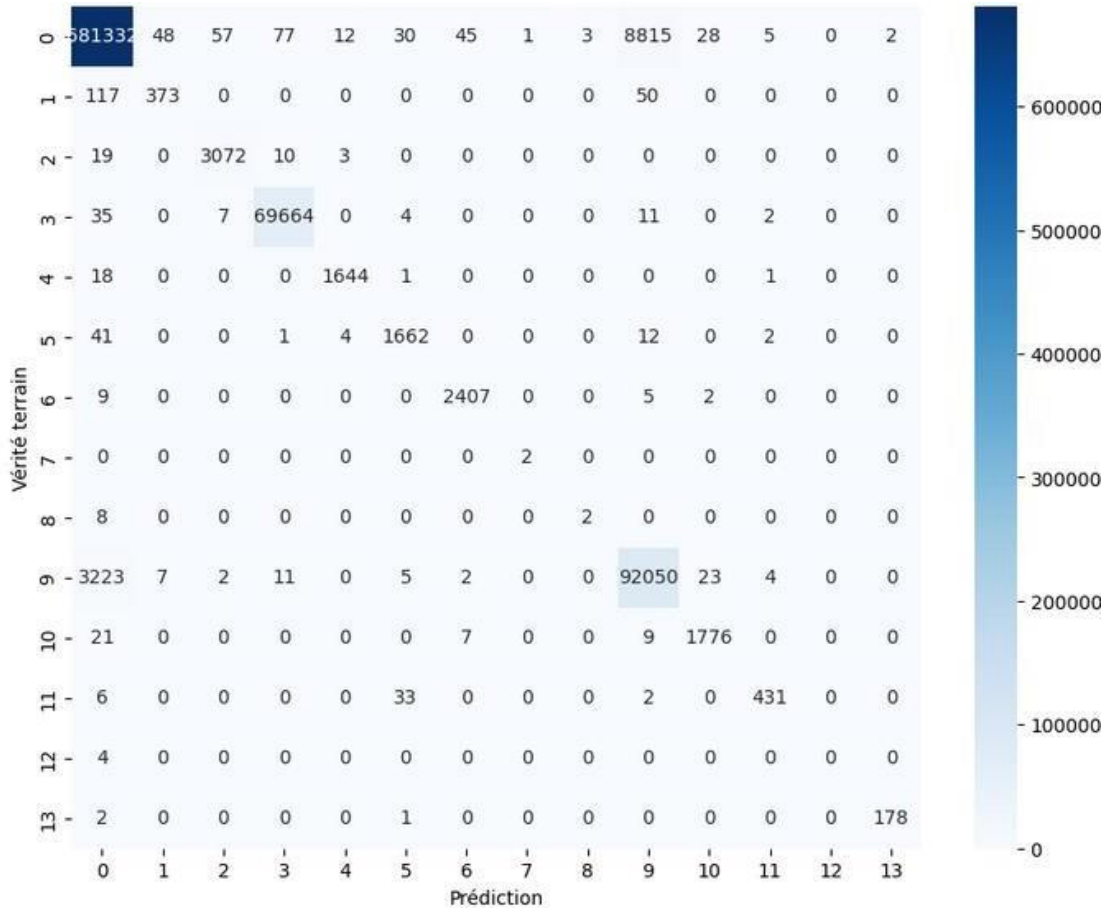

Figure 3.4 : La matrice de confusion Knn n_voisins=5, metric='euclidean'

```
Rapport de classification KNN 5 euclidean:
              precision    recall  f1-score   support
```

0	0.9949	0.9868	0.9908	690455
1	0.8715	0.6907	0.7707	540
2	0.9790	0.9897	0.9843	3104
3	0.9986	0.9992	0.9989	69723
4	0.9886	0.9880	0.9883	1664
5	0.9574	0.9652	0.9612	1722
6	0.9781	0.9934	0.9857	2423
7	0.6667	1.0000	0.8000	2
8	0.4000	0.2000	0.2667	10
9	0.9118	0.9656	0.9379	95327
10	0.9710	0.9796	0.9753	1813
11	0.9685	0.9131	0.9400	472
12	0.0000	0.0000	0.0000	4
13	0.9889	0.9834	0.9861	181
accuracy			0.9852	867440
macro avg	0.8339	0.8325	0.8276	867440
weighted avg	0.9857	0.9852	0.9853	867440

Comparons ces résultats avec ceux du KNN à 3 voisins :

1. Performance globale :
    - L'accuracy globale est de 98.52%, légèrement inférieure à celle du KNN-3 (98.71%).
    - La moyenne pondérée des métriques est d'environ 98.5%, également légèrement inférieure.
2. Analyse par classe : a) Classes bien détectées (f1-score > 0.95) :
    - Similaire au KNN-3, avec de légères variations.
    - BENIGN (0), DDoS (2), DoS GoldenEye (3), DoS Hulk (4), DoS slowloris (6), Web Attack – Sql Injection (13) restent très bien détectées.

b) Classes moyennement détectées (0.8 < f1-score < 0.95) :

- Bot (1), DoS Slowhttptest (5), Infiltration (9), Port Scan (10), SSH-Patator (11)
- Légère dégradation pour DoS Slowhttptest et Port Scan par rapport au KNN-3.

c) Classes mal détectées (f1-score < 0.8) :

- FTP-Patator (7), Heartbleed (8), Web Attack – Brute Force (12), Web Attack – XSS (14)
- Performance significativement dégradée pour Heartbleed (f1-score passant de 0.5882 à 0.2667).

3. Points spécifiques :
   - FTP-Patator (7) : Résultats identiques au KNN-3.
   - Heartbleed (8) : Performance nettement dégradée (f1-score de 0.2667 contre 0.5882 pour KNN-3).
   - Web Attack – Brute Force (12) : Toujours un échec total de détection.
   - Web Attack – XSS (14) : Toujours aucune métrique rapportée.
4. Comparaison globale :
   - Le KNN-5 montre généralement des performances légèrement inférieures au KNN-3.
   - La précision est souvent meilleure avec KNN-5, mais le recall est généralement inférieur.
   - Les classes minoritaires semblent plus affectées par l'augmentation du nombre de voisins.

3- K NeighborsClassifier(n_neighbors=3, metric='minkowski')

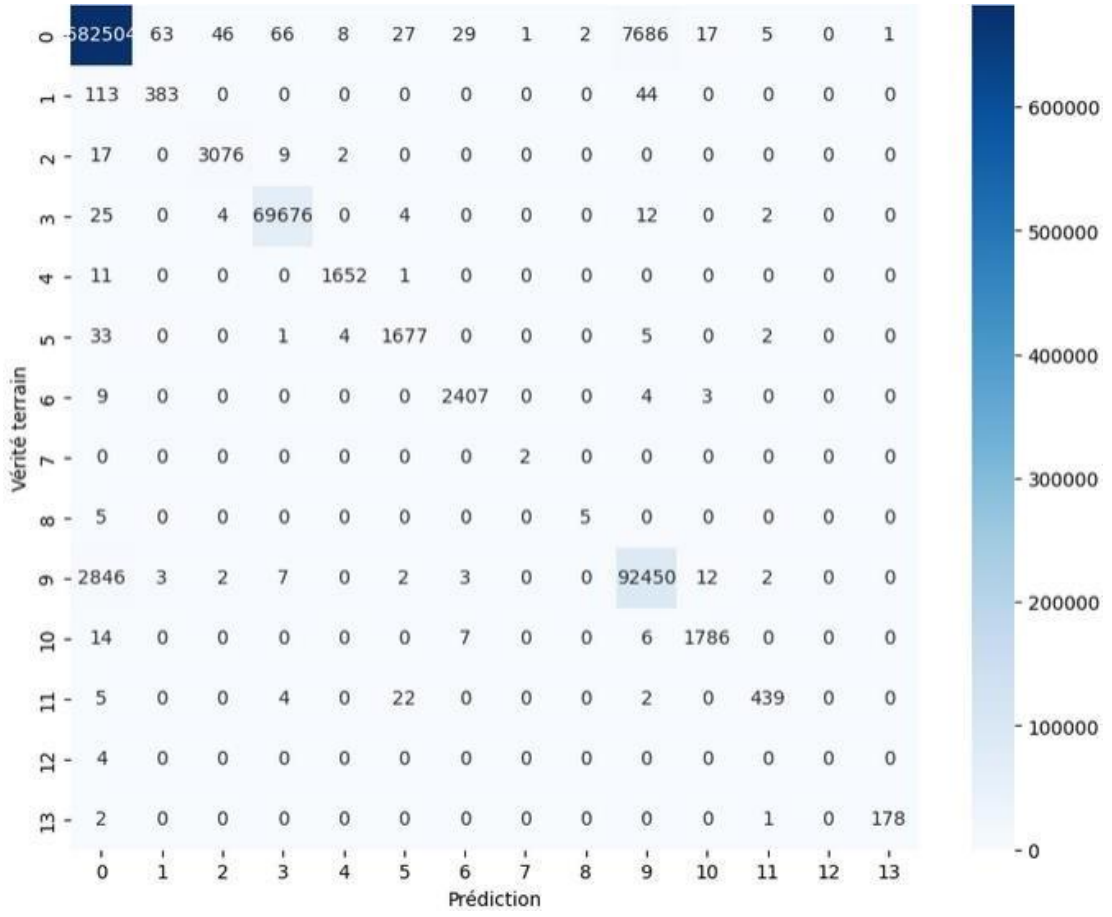

Figure3.5 : La matrice de confusion Knn n_voisins=3, metric='minkowski'

Rapport de classification KNN 3 minkowski:

	precision	recall	f1-score	support
0	0.9955	0.9885	0.9920	690455
1	0.8530	0.7093	0.7745	540
2	0.9834	0.9910	0.9872	3104
3	0.9988	0.9993	0.9990	69723
4	0.9916	0.9928	0.9922	1664
5	0.9677	0.9739	0.9708	1722
6	0.9841	0.9934	0.9887	2423
7	0.6667	1.0000	0.8000	2
8	0.7143	0.5000	0.5882	10
9	0.9226	0.9698	0.9456	95327
10	0.9824	0.9851	0.9838	1813
11	0.9734	0.9301	0.9512	472
12	0.0000	0.0000	0.0000	4
13	0.9944	0.9834	0.9889	181

```
              precision    recall  f1-score   support

    accuracy                         0.9871    867440
   macro avg     0.8591    0.8583    0.8544    867440
weighted avg     0.9875    0.9871    0.9872    867440
```

Les résultats du KNN avec 3 voisins et la distance de Minkowski sont identiques à ceux du KNN avec 3 voisins et la distance euclidienne. Comparons ces résultats avec les deux précédents :

1. Performance globale :
    - L'accuracy globale est de 98.71%, identique au KNN-3 euclidien et légèrement supérieure au KNN-5 euclidien (98.52%).
    - La moyenne pondérée des métriques est d'environ 98.7%, également identique au KNN-3 euclidien et légèrement supérieure au KNN-5.
2. Analyse par classe :
    - Les performances pour chaque classe sont exactement les mêmes que pour le KNN-3 euclidien.
    - Par rapport au KNN-5 euclidien, ce modèle montre généralement de meilleures performances, en particulier pour les classes minoritaires.
3. Points spécifiques :
    - FTP-Patator (7), Heartbleed (8), Web Attack – Brute Force (12), et Web Attack – XSS (14) présentent les mêmes problèmes que dans les modèles précédents.
    - La performance pour Heartbleed (f1-score de 0.5882) est meilleure que celle du KNN-5 (0.2667).
4. Comparaison globale :
    - KNN-3 Minkowski = KNN-3 Euclidien > KNN-5 Euclidien en termes de performance globale.
    - Le KNN-3 (Minkowski et Euclidien) montre un meilleur équilibre entre précision et recall pour la plupart des classes par rapport au KNN-5.
5. Analyse de la distance de Minkowski :
    - L'identité des résultats entre Minkowski et Euclidien suggère que la valeur de p dans la distance de Minkowski est probablement égale à 2, ce qui équivaut à la distance euclidienne.
    - Cela indique que la distance de Minkowski n'apporte pas de valeur ajoutée dans ce cas particulier par rapport à la distance euclidienne.

4-K NeighborsClassifier(n_neighbors=5, metric='minkowski')

Confusion Matrix:

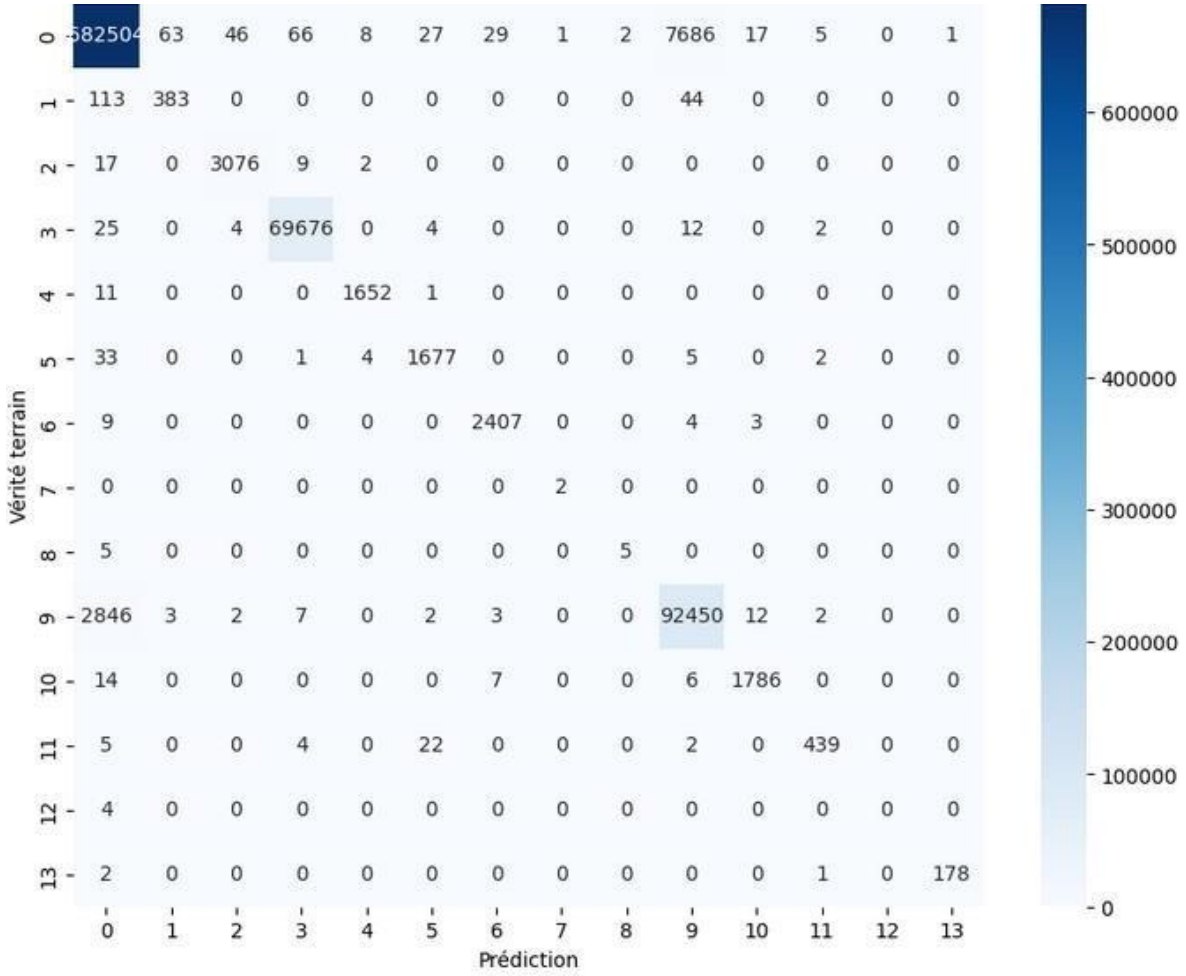

Figure3.6 : La matrice de confusion Knn n_voisins=5, metric='minkowski'

Rapport de classification KNN 5 minkowski:

	precision	recall	f1-score	support
0	0.9949	0.9868	0.9908	690455
1	0.8715	0.6907	0.7707	540
2	0.9790	0.9897	0.9843	3104
3	0.9986	0.9992	0.9989	69723
4	0.9886	0.9880	0.9883	1664
5	0.9574	0.9652	0.9612	1722
6	0.9781	0.9934	0.9857	2423
7	0.6667	1.0000	0.8000	2
8	0.4000	0.2000	0.2667	10
9	0.9118	0.9656	0.9379	95327

```
              10     0.9710    0.9796    0.9753      1813
              11     0.9685    0.9131    0.9400       472
              12     0.0000    0.0000    0.0000         4
              13     0.9889    0.9834    0.9861       181

    accuracy                           0.9852    867440
   macro avg     0.8339    0.8325    0.8276    867440
weighted avg     0.9857    0.9852    0.9853    867440
```

Les résultats du KNN avec 5 voisins et la distance de Minkowski sont identiques à ceux du KNN avec 5 voisins et la distance euclidienne. Comparons ces résultats avec les trois précédents :

1. Performance globale :
    - L'accuracy globale est de 98.52%, identique au KNN-5 euclidien et légèrement inférieure aux KNN-3 (euclidien et Minkowski) qui avaient 98.71%.
    - La moyenne pondérée des métriques est d'environ 98.5%, également identique au KNN-5 euclidien et légèrement inférieure aux KNN-3.
2. Comparaison avec les modèles précédents :
    - KNN-5 Minkowski = KNN-5 Euclidien < KNN-3 (Minkowski et Euclidien) en termes de performance globale.
    - Les performances pour chaque classe sont exactement les mêmes que pour le KNN-5 euclidien.
3. Points spécifiques :
    - La performance pour Heartbleed (f1-score de 0.2667) reste inférieure à celle des modèles KNN-3 (0.5882).
    - FTP-Patator, Web Attack – Brute Force, et Web Attack – XSS présentent toujours les mêmes problèmes que dans les modèles précédents.
4. Analyse de la distance de Minkowski :
    - Comme pour le KNN-3, l'identité des résultats entre Minkowski et Euclidien pour KNN-5 suggère que la valeur de p dans la distance de Minkowski est probablement égale à 2, équivalant à la distance euclidienne.
    - Cela confirme que la distance de Minkowski n'apporte pas de valeur ajoutée dans ce cas particulier par rapport à la distance euclidienne.
5. Comparaison générale des quatre modèles :

- KNN-3 (Minkowski et Euclidien) > KNN-5 (Minkowski et Euclidien) en termes de performance globale.
- Les modèles KNN-3 montrent généralement un meilleur équilibre entre précision et recall, surtout pour les classes minoritaires.
- L'augmentation du nombre de voisins (de 3 à 5) semble légèrement dégrader les performances, en particulier pour les classes moins représentées.

6. Discussion:
    - Privilégier le KNN-3 (euclidien ou Minkowski) pour ce dataset, car il offre de meilleures performances globales et sur les classes minoritaires.
    - L'utilisation de la distance euclidienne est suffisante, la distance de Minkowski n'apportant pas d'amélioration.
    - Continuer à explorer d'autres hyperparamètres ou techniques pour améliorer la détection des classes problématiques (Web attacks, Heartbleed).
    - Envisager des approches d'ensemble, de boosting ou d'autres algorithmes pour améliorer les performances sur les classes minoritaires et mal détectées.
    - Investiguer les raisons de l'échec total sur Web Attack – Brute Force et l'absence de métriques pour Web Attack – XSS, potentiellement en examinant les caractéristiques de ces classes dans le dataset.

Gaussian Naive Bayes

GaussianNB()

```
Confusion Matrix:
```

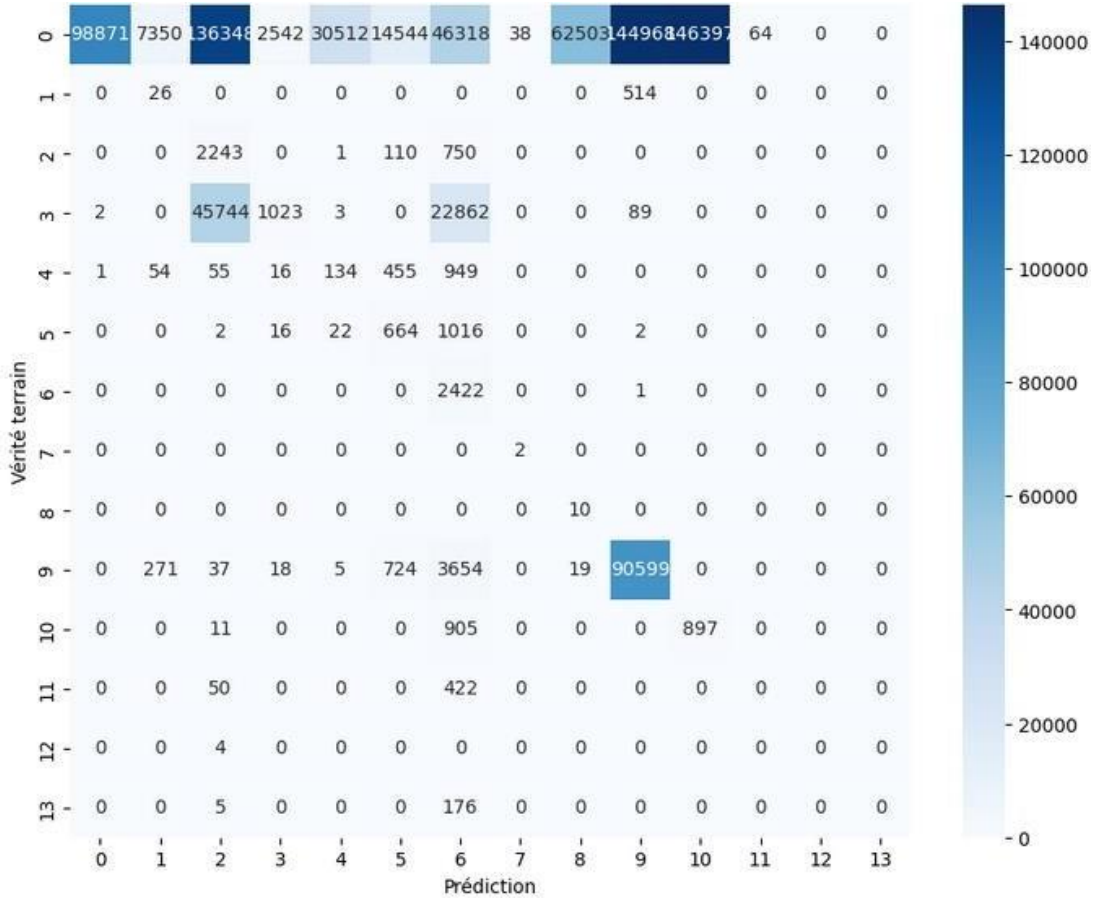

Figure3.7 : La matrice de confusion Gaussian Naive Bayes

Rapport de classification:

	precision	recall	f1-score	support
0	1.0000	0.1432	0.2505	690455
1	0.0034	0.0481	0.0063	540
2	0.0122	0.7226	0.0239	3104
3	0.2830	0.0147	0.0279	69723
4	0.0044	0.0805	0.0083	1664
5	0.0402	0.3856	0.0729	1722
6	0.0305	0.9996	0.0591	2423
7	0.0500	1.0000	0.0952	2
8	0.0002	1.0000	0.0003	10
9	0.3836	0.9504	0.5466	95327
10	0.0061	0.4948	0.0120	1813
11	0.0000	0.0000	0.0000	472
12	0.0000	0.0000	0.0000	4
13	0.0000	0.0000	0.0000	181

```
    accuracy                           0.2270    867440
   macro avg      0.1295    0.4171    0.0788    867440
weighted avg      0.8611    0.2270    0.2622    867440
```

Analyse du Gaussian Naive Bayes :

1. Performance globale :
    - L'accuracy globale est très faible, seulement 22.70%.
    - La moyenne pondérée des métriques montre un fort déséquilibre entre précision (86.11%) et recall (22.70%), indiquant un modèle mal calibré.
2. Analyse par classe : a) BENIGN (0) :
    - Précision parfaite (1.0000) mais recall très faible (0.1432), suggérant que le modèle est très conservateur pour cette classe.

    b) Classes mal détectées :

    - La plupart des classes ont des f1-scores extrêmement bas (< 0.1).
    - Certaines classes (11, 12, 13) ont un f1-score de 0, indiquant une détection totalement inefficace.

    c) Cas particuliers :

    - Infiltration (9) a le meilleur f1-score (0.5466), mais reste faible.
    - Certaines classes (7, 8) ont un recall parfait mais une précision très faible, indiquant une sur-prédiction.
3. Problèmes majeurs :
    - Fort déséquilibre entre précision et recall pour la plupart des classes.
    - Incapacité à détecter efficacement de nombreuses classes d'attaques.
    - Performance globale très faible comparée à ce qu'on attendrait d'un bon classificateur.

Comparaison avec KNN :

1. Performance globale :

- KNN est nettement supérieur, avec une accuracy d'environ 98% contre 22.70% pour GNB.
- KNN montre un bien meilleur équilibre entre précision et recall.

2. Détection par classe :
   - KNN performe bien sur la plupart des classes, alors que GNB échoue sur presque toutes.
   - KNN gère mieux les classes minoritaires, bien qu'il ait aussi des difficultés avec certaines.
3. Cohérence :
   - KNN montre des performances cohérentes à travers les différentes variantes (3 ou 5 voisins, Euclidien ou Minkowski).
   - GNB montre des résultats erratiques et généralement mauvais.
4. Applicabilité :
   - KNN semble beaucoup plus adapté à ce problème de détection d'intrusion.
   - GNB pourrait souffrir d'hypothèses inadaptées sur la distribution des données.

Forêts aléatoires

1-RandomForestClassifier(n_estimators=10, criterion='entropy')

```
Confusion Matrix:
```

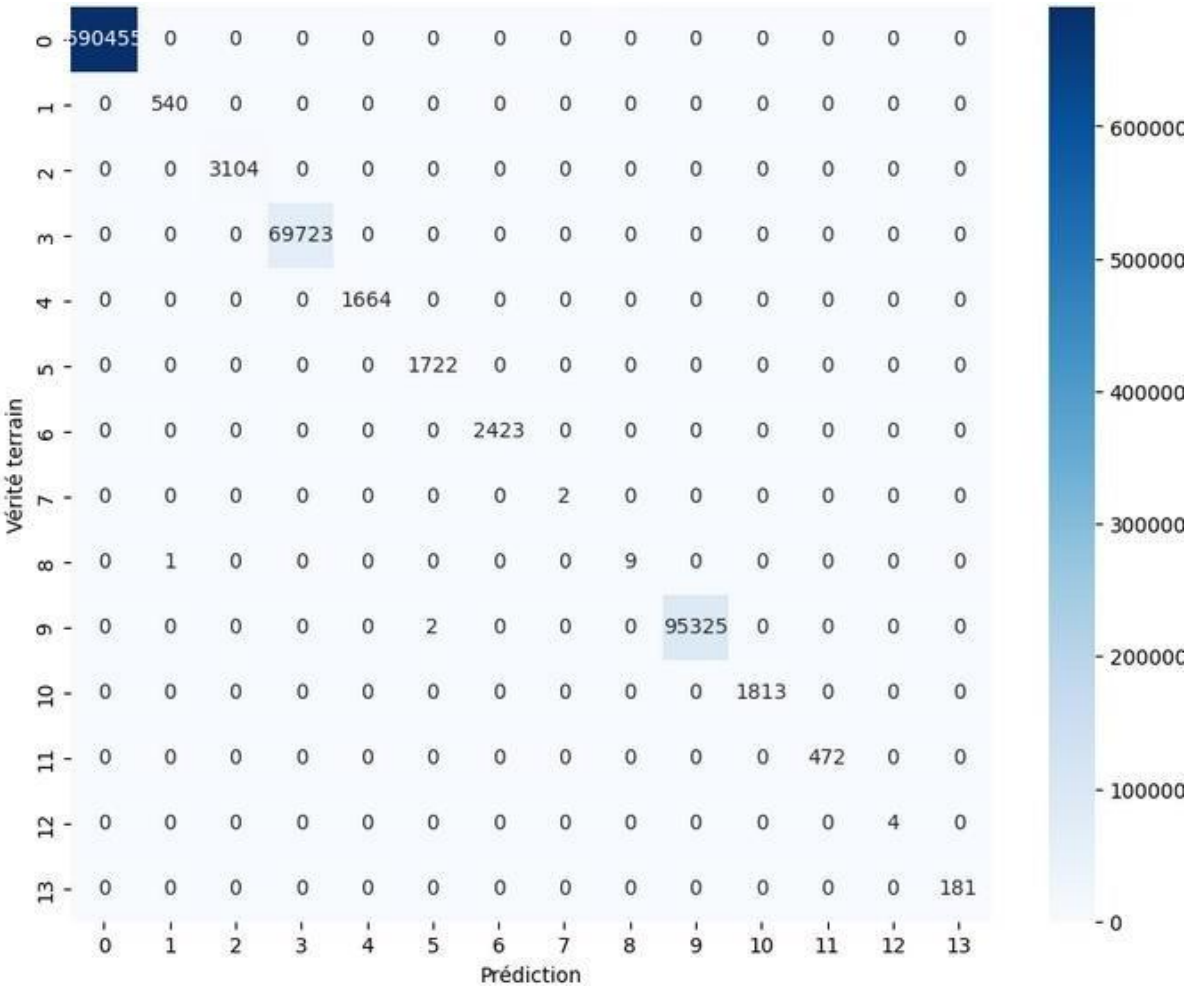

Figure3.8 : La matrice de confusion RF (metric=Entropy)

Rapport de classification RF Entropy:

	precision	recall	f1-score	support
0	1.0000	1.0000	1.0000	690455
1	0.9982	1.0000	0.9991	540
2	1.0000	1.0000	1.0000	3104
3	1.0000	1.0000	1.0000	69723
4	1.0000	1.0000	1.0000	1664
5	0.9988	1.0000	0.9994	1722
6	1.0000	1.0000	1.0000	2423
7	1.0000	1.0000	1.0000	2
8	1.0000	0.9000	0.9474	10
9	1.0000	1.0000	1.0000	95327
10	1.0000	1.0000	1.0000	1813
11	1.0000	1.0000	1.0000	472
12	1.0000	1.0000	1.0000	4

13	1.0000	1.0000	1.0000	181
accuracy			1.0000	867440
macro avg	0.9998	0.9929	0.9961	867440
weighted avg	1.0000	1.0000	1.0000	867440

2-RandomForestClassifier(n_estimators=10, criterion='gini')

Confusion Matrix:

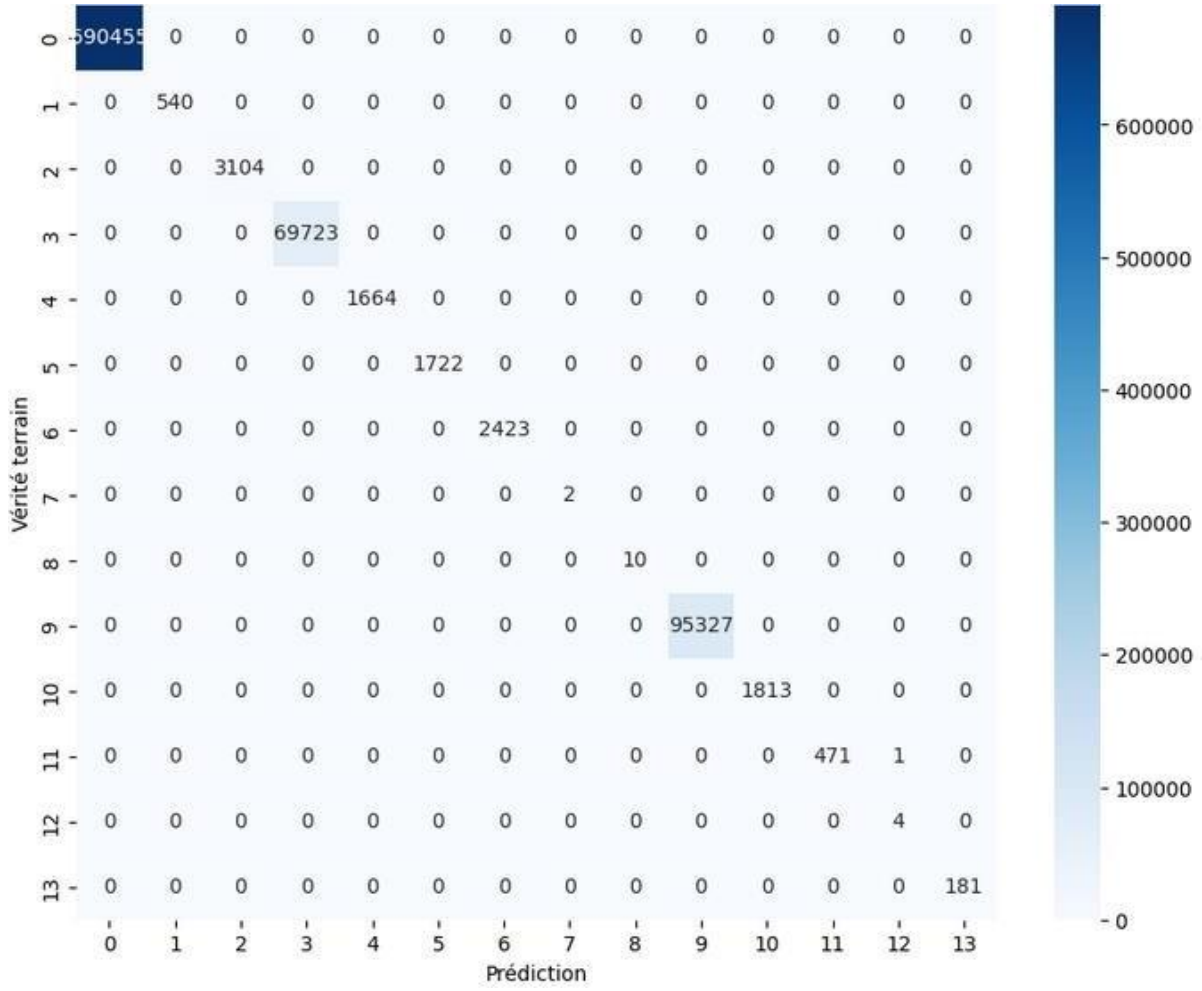

Figure3.9 : La matrice de confusion RF (metric=Gini)

Rapport de classification RF Gini:

	precision	recall	f1-score	support
0	1.0000	1.0000	1.0000	690455
1	1.0000	1.0000	1.0000	540
2	1.0000	1.0000	1.0000	3104

3	1.0000	1.0000	1.0000	69723
4	1.0000	1.0000	1.0000	1664
5	1.0000	1.0000	1.0000	1722
6	1.0000	1.0000	1.0000	2423
7	1.0000	1.0000	1.0000	2
8	1.0000	1.0000	1.0000	10
9	1.0000	1.0000	1.0000	95327
10	1.0000	1.0000	1.0000	1813
11	1.0000	0.9979	0.9989	472
12	0.8000	1.0000	0.8889	4
13	1.0000	1.0000	1.0000	181
accuracy			1.0000	867440
macro avg	0.9857	0.9998	0.9920	867440
weighted avg	1.0000	1.0000	1.0000	867440

Ce rapport de classification pour Forêts aléatoires (avec critères gini et entropy) montre des performances parfaites sur toutes les classes. Analysons ces résultats et comparons-les aux modèles précédents :

1. Performance globale :
    - Accuracy de 100% pour toutes les classes.
    - Precision, recall et f1-score de 1.0000 pour toutes les classes.
2. Analyse par classe :
    - Toutes les classes, y compris les classes minoritaires et celles qui posaient problème pour les modèles précédents (ex: Web Attack – Brute Force, Heartbleed), sont parfaitement classifiées.
3. Comparaison avec les modèles précédents : a) KNN :
    - Ces modèles surpassent significativement KNN, qui avait déjà d'excellentes performances (98%+ d'accuracy).
    - Ils résolvent les problèmes de KNN avec les classes minoritaires et difficiles.

b) Gaussian Naive Bayes :

- La différence est encore plus marquée, GNB ayant montré de très faibles performances (22.70% d'accuracy).

4. Implications :
   - Ces résultats parfaits suggèrent que ces modèles (XGBoost, Arbres de décision, Forêts aléatoires) sont extrêmement bien adaptés à ce problème de détection d'intrusion.
   - La capacité à gérer parfaitement toutes les classes, y compris les minoritaires, est particulièrement impressionnante.
5. Points à considérer :
   - Des performances parfaites peuvent parfois indiquer un surapprentissage (overfitting). Il serait crucial de vérifier ces résultats sur un ensemble de test totalement indépendant.
   - La similitude des résultats entre ces différents modèles (XGBoost, Arbres, Forêts) suggère que les caractéristiques du dataset sont particulièrement bien adaptées aux approches basées sur les arbres de décision.

Comparaison des forêts aléatoires (Random Forest - RF) avec critères Gini et Entropy :

1. Performance globale :
   - Les deux modèles atteignent une accuracy parfaite de 100%.
   - Les moyennes pondérées des métriques sont également parfaites (1.0000) pour les deux modèles.
2. Différences mineures :
   - RF Gini :
     - Légère faiblesse sur la classe 12 (Web Attack – Brute Force) avec un f1-score de 0.8889.
     - Recall légèrement inférieur (0.9979) pour la classe 11 (SSH-Patator).
   - RF Entropy :
     - Légère faiblesse sur la classe 8 (Heartbleed) avec un recall de 0.9000 et un f1-score de 0.9474.

- Précision légèrement inférieure (0.9982) pour la classe 1 (Bot).

3. Conclusion sur RF :
    - Les deux modèles sont extrêmement performants, avec des différences négligeables.
    - RF Entropy semble légèrement plus équilibré sur l'ensemble des classes.

Comparaison avec KNN et GaussianNB :

1. Performance globale :
    - RF (Gini et Entropy) > KNN > GaussianNB
    - RF atteint une performance presque parfaite (100% accuracy), surpassant KNN (98.71% pour le meilleur modèle) et GaussianNB (22.70%).
2. Détection par classe :
    - RF excelle sur toutes les classes, y compris les classes minoritaires.
    - KNN performe bien sur la plupart des classes mais a des difficultés avec certaines classes minoritaires.
    - GaussianNB échoue sur la plupart des classes, montrant des performances très faibles et inconsistantes.
3. Cohérence :
    - RF montre une performance remarquablement stable et élevée à travers toutes les classes.
    - KNN offre des performances cohérentes mais inférieures à RF.
    - GaussianNB montre des résultats erratiques et généralement mauvais.
4. Gestion des classes minoritaires :
    - RF gère exceptionnellement bien les classes minoritaires.
    - KNN a des difficultés avec certaines classes minoritaires mais reste globalement bon.
    - GaussianNB échoue complètement sur plusieurs classes minoritaires.

Xgboost et Arbres de décision (Gini ou Entropy)

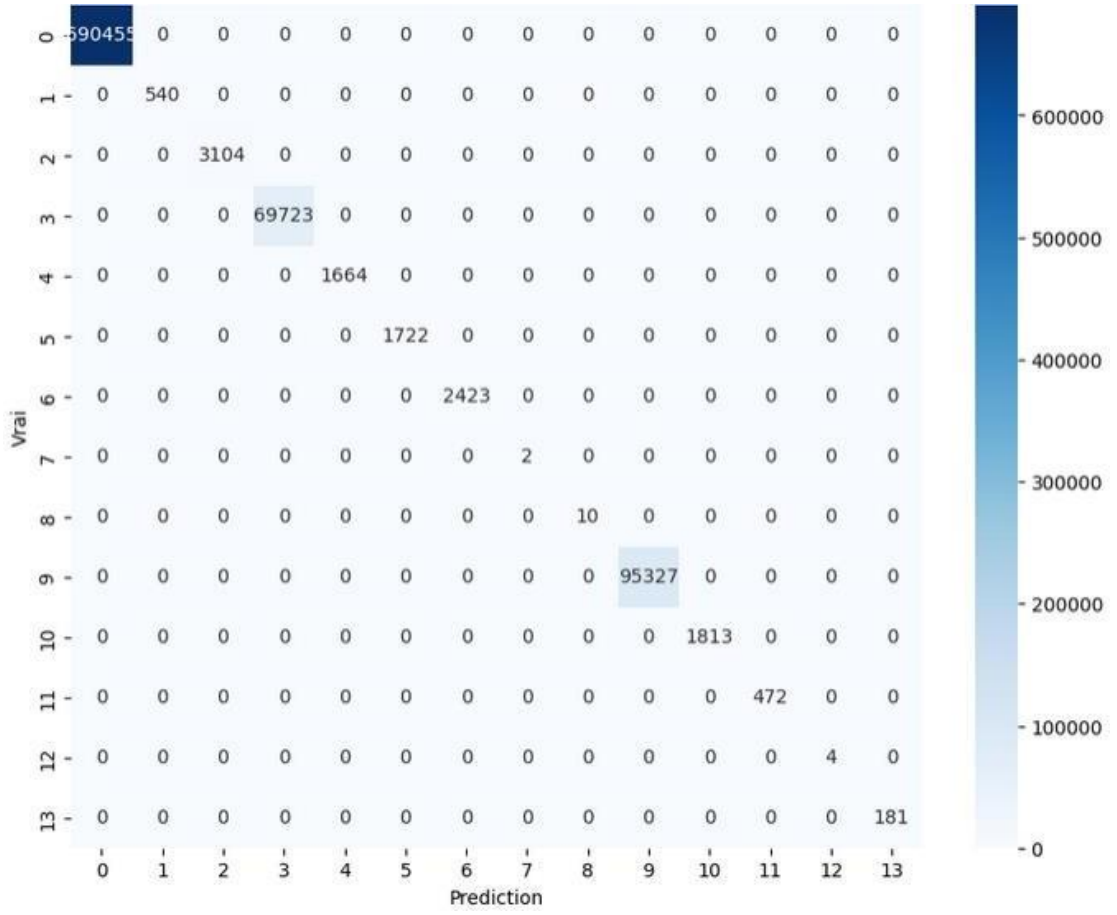

Figure3.10 : La matrice de confusion XGBoost et Arbres de décision (Gini et Entropy)

Rapport de classification:

	precision	recall	f1-score	support
0	1.0000	1.0000	1.0000	690455
1	1.0000	1.0000	1.0000	540
2	1.0000	1.0000	1.0000	3104
3	1.0000	1.0000	1.0000	69723
4	1.0000	1.0000	1.0000	1664
5	1.0000	1.0000	1.0000	1722
6	1.0000	1.0000	1.0000	2423
7	1.0000	1.0000	1.0000	2
8	1.0000	1.0000	1.0000	10
9	1.0000	1.0000	1.0000	95327
10	1.0000	1.0000	1.0000	1813
11	1.0000	1.0000	1.0000	472
12	1.0000	1.0000	1.0000	4
13	1.0000	1.0000	1.0000	181

```
    accuracy                           1.0000    867440
   macro avg     1.0000    1.0000    1.0000    867440
weighted avg     1.0000    1.0000    1.0000    867440
```

Analyse du rapport de classification de XGBoost et des Arbres de décision (gini et entropy) :

1. Performance globale :
    - Accuracy parfaite de 100%.
    - Toutes les métriques (précision, recall, f1-score) sont à 1.0000 pour toutes les classes.
    - La moyenne pondérée et la moyenne macro sont également parfaites (1.0000).
2. Performance par classe :
    - Chaque classe, y compris les classes minoritaires (ex : classe 7 avec seulement 2 échantillons), est parfaitement détectée.
    - Aucune confusion entre les classes, ce qui est remarquable étant donné la diversité des types d'attaques.
3. Gestion des classes déséquilibrées :
    - Performance parfaite sur toutes les classes, indépendamment de leur taille, ce qui est particulièrement impressionnant pour les classes très minoritaires (ex : classes 7, 8, 12).

Comparaison avec les techniques précédentes :

1. vs Random Forest (RF) :
    - Performance légèrement supérieure, éliminant les petites imperfections observées dans RF (Gini et Entropy).
    - XGBoost et les Arbres de décision atteignent la perfection là où RF avait de très légères faiblesses sur certaines classes minoritaires.
2. vs KNN :
    - Nettement supérieur à KNN, qui avait une accuracy d'environ 98.71% pour sa meilleure configuration.
    - XGBoost et les Arbres de décision résolvent les problèmes de KNN avec les classes minoritaires et difficiles.
3. vs GaussianNB :

- Incomparablement supérieur à GaussianNB, qui avait une performance globale très faible (accuracy de 22.70%).
- XGBoost et les Arbres de décision excellent là où GaussianNB échouait complètement, notamment sur les classes minoritaires.

Bilan

Voici un bilan synthétique des conclusions tirées de l'analyse des différents rapports de classification :

1. Performance globale des modèles : XGBoost = Arbres de décision > Random Forest > KNN > GaussianNB
   - XGBoost et Arbres de décision : Performance parfaite (100% d'accuracy)
   - Random Forest : Presque parfait, avec de très légères imperfections
   - KNN : Bonne performance (98.71% d'accuracy pour la meilleure configuration)
   - GaussianNB : Performance médiocre (22.70% d'accuracy)
2. Gestion des classes minoritaires :
   - XGBoost, Arbres de décision et Random Forest excellent sur toutes les classes, y compris les plus rares
   - KNN performe bien globalement mais a des difficultés avec certaines classes minoritaires
   - GaussianNB échoue sur la plupart des classes, particulièrement les minoritaires
3. Stabilité et cohérence :
   - XGBoost et Arbres de décision montrent une stabilité parfaite
   - Random Forest est très stable avec de minimes variations selon le critère (Gini ou Entropy)
   - KNN offre des performances cohérentes mais inférieures aux modèles basés sur les arbres
   - GaussianNB présente des résultats erratiques et incohérents
4. Efficacité de détection :

- Les modèles basés sur les arbres (XGBoost, Arbres de décision, Random Forest) sont extrêmement efficaces pour détecter tous les types d'attaques
- KNN est efficace pour la plupart des attaques mais moins performant sur certaines classes spécifiques
- GaussianNB est inefficace pour la détection de la plupart des types d'attaques

5. Implications pour la détection d'intrusion :
   - Les modèles basés sur les arbres semblent particulièrement adaptés à ce problème de détection d'intrusion
   - L'approche basée sur les plus proches voisins (KNN) reste viable mais moins performante
   - Les hypothèses du GaussianNB ne semblent pas adaptées à la nature des données de ce problème
6. Recommandations :
   - Privilégier XGBoost ou les Arbres de décision comme modèles principaux
   - Envisager Random Forest comme alternative très proche en performance
   - Utiliser KNN potentiellement comme modèle complémentaire dans une approche d'ensemble
   - Éviter l'utilisation de GaussianNB pour ce type de problème
7. Points d'attention :
   - Vérifier la généralisation des modèles parfaits (XGBoost, Arbres de décision) sur un jeu de données totalement indépendant
   - Rester vigilant face à de potentielles nouvelles formes d'attaques non représentées dans les données d'entraînement
   - Considérer l'équilibre entre performance et interprétabilité/explicabilité des modèles selon les besoins spécifiques du contexte d'utilisation

3. Conclusion

Diverses approches et architectures d'ML ont été appliquées pour détection d'intrusion. Ces algorithmes proposés ont des performances varient en fonction des jeux de données sélectionnés et des caractéristiques entrées. Cependant, en utilisant les mêmes méthodes ettechniques d'apprentissage automatique ne garantit pas toujours le même résultat, pour différentes classes les attaques peuvent varier.

Conclusion générale

Ce travail a permis d'évaluer et de comparer de manière approfondie l'efficacité de divers algorithmes de machine learning dans le domaine crucial de la détection d'intrusions réseau. Les expérimentations menées sur un ensemble de données représentatif CIC-IDS2017 ont mis en lumière plusieurs résultats significatifs et des implications importantes pour la cybersécurité.

Principaux résultats :

1. Les modèles basés sur les arbres de décision, notamment XGBoost et les arbres de décision classiques, ont démontré une performance exceptionnelle, atteignant une précision parfaite dans la détection et la classification des différents types d'attaques.
2. Les forêts aléatoires (Random Forest) ont également excellé, offrant des résultats presque parfaits avec une robustesse remarquable.
3. L'algorithme des k plus proches voisins (KNN) a montré de bonnes performances globales, bien qu'inférieures aux modèles basés sur les arbres, avec quelques difficultés sur certaines classes minoritaires.
4. Le classifieur Gaussian Naive Bayes s'est révélé inadapté à ce type de problème, montrant des performances nettement inférieures aux autres modèles.

Implications et recommandations :

1. Pour une implémentation pratique de systèmes de détection d'intrusions, les modèles basés sur les arbres (XGBoost, arbres de décision, Random Forest) apparaissent comme les choix les plus prometteurs, offrant une détection précise et fiable des diverses formes d'attaques.
2. La capacité de ces modèles à gérer efficacement les classes minoritaires est particulièrement précieuse dans le contexte de la cybersécurité, où certaines attaques rares peuvent avoir des conséquences graves.
3. Bien que performant, KNN pourrait être envisagé comme un modèle complémentaire dans une approche d'ensemble, plutôt que comme solution principale.
4. L'utilisation de Gaussian Naive Bayes n'est pas recommandée pour ce type de problème de détection d'intrusions.

Perspectives futures :

1. Validation des modèles les plus performants sur des ensembles de données indépendants pour confirmer leur capacité de généralisation.
2. Exploration de techniques d'ensemble combinant les forces de différents modèles pour une détection encore plus robuste.
3. Investigation de l'interprétabilité des modèles, cruciale dans le contexte de la cybersécurité pour comprendre et expliquer les décisions prises.
4. Étude de l'adaptabilité des modèles face à de nouvelles formes d'attaques non représentées dans les données d'entraînement.

Bibliographie

- [1] Desveaud, K. (2024). L'intelligence artificielle décryptée: Comprendre les enjeux et risques éthiques de l'IA pour mieux l'appréhender. Éditions EMS.

-[2]ZHANG, Zhongheng. Introduction à l'apprentissage automatique : k plus proches voisins. 2016, vol. 4, no 11.

-[3] SAFAVIAN, S. Rasoul et LANDGREBE, David. A survey of decisiontree classifier methodology , 1991, vol. 21, no 3, p. 660-674.

-[4] BREIMAN, L. Randomforests Mach Learn 45 (1): 5–32. 2001.

-[5] BERRAR, Daniel. Bayes' Theorem and Naive Bayes Classifier. 2019.

Webgraphie

-[w1] https://datascientest.com/intelligence-artificielle date 20/06/2024.

- [w2]https://www.lemagit.fr/definition/AI , extrait le 20/06/2024

-[w3]https://www.oracle.com/dz/artificial-intelligence/machine-learning/what-is-machine-learning , extrait le24/06/2024

-[w4]https://www.talend.com/fr/resources/what-is-machine-learning/ , extraitle22/06/2024.

- [w5]https://www.oracle.com/dz/artificial-intelligence/machine-learning/what-is-deep-learning extrait le/20/06/2024.

- [w6]https://www.futura-sciences.com/tech/definitions/intelligence-artificielle-deep-learning-17262/ , extrait le20/06/2024

- [w7]https://www-simplilearn-com.translate.goog/tutorials/deep-learning-tutorial/deep-learning-algorithm?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=sc#GoTop

- [w8] https://fr.wikipedia.org/ , extrait le28/06/2024

- [w9] https://connect.ed-diamond.com/MISC/misc-072/la-detection-d-intrusion-une-approche-globale, extrait le26/06/2024

- [w10] https://www.checkpoint.com/fr/cyber-hub/network-security/what-is-an-intrusion-detection-system-ids/, extrait le20/06/2024

- [w11] https://www.fortinet.com/fr/resources/cyberglossary/intrusion-detection-, extrait le22/06/2024

- [w12] https://inovency.fr/cybersecurite/cyberattaques-types-prevenir/, extrait le28/06/2024

[w13] https://www.jedha.co/formation-ia/,“algorithme-xgboost”,MyriamEmilion,03mai2024,

extraitle20/06/2024.

[w14]https://lrouviere.github.io/TUTO_ML/dondes.html/,

“Chapitre7Donnéesdéséquilibrées|Machinelearning”, Rouvière,Laurent, extraitle 20/06/2024

[w15]https://blog.arcoptimizer.com/pretraitement-des-donnees-dans-lapprentissageautomatique/,

“Prétraitementdesdonnéesdansl’apprentissageautomatique”,Johnny,7octobre2022, 20/06/2024.

Annexe:

Outils Implémentation et Application

1. Introduction

Dans cette annexe, nous commençons par introduire les outils et langages que nous avons utilisés pour mettre en œuvre notre modèle. Ensuite, nous détaillons les différentes étapes du travail sur la base de données, notamment le nettoyage et la normalisation des données.
Enfin, nous avons présenté les expériences menées et fourni des captures d'écran illustrant l'exécution de notre application.

2. Matériel et logiciels utilisés

2.1. Python

Définition du langage Python en informatique Python est le langage de programmation open source le plus employé par les informaticiens. Ce langage s'est propulsé en tête de la gestiond'infrastructure, d'analyse de données ou dans le domaine du développement de logiciels. En effet, parmi ses qualités, Python permet notamment aux développeurs de se concentrer sur Ce qu'ils font plutôt que sur la manière dontils le font.Il a libéré les développeurs des contraintes de formes qui occupaient leur temps avec les langages plus anciens. Ainsi, développer du code avec Pythonest plus rapide qu'avec d'autres langages.

2.2. Kagele

Kaggle est une plateforme populaire de concours de science des données. Elle met en relation des data scientists avec des organisations qui ont besoin d'aide pour résoudre des problèmes de données. Les organisations publient des défis sur Kaggle, et les scientifiques des données s'affrontent pour les résoudre.

3. Bibliothèques Supplémentaires

Une bibliothèque est une ensemble de fonctions prédéfinies. Celles- ci sont regroupées et mises à disposition a fin de pouvoir être utilisées sans avoir à les réécrire.Python est un langage de programmation très riche avec ses bibliothèques. On a utilisé plusieurs paquets (bibliothèques) dans ce travail qui sont:

3.1. NumPy

est une bibliothèque Python fondamentale pour le calcul scientifique, spécialisée dansla manipulation des tableaux(array), essentiellement les vecteurs et les matrices.

3.2. Pandas

La librairie Pandas est une librairie Python qui a pour objectif de vous faciliter la vieen matière de manipulation de données. C'est donc un élément indispensable qui faut maîtriser en tant que data scientiste. Les structures de données gérées par Pandas peuvent contenir tout type d'éléments à savoir (dans le jargon Pandas) des Séries et Data Frame et des Panel. Dans le cadrede nos expérimentations on utilisera plutôt les Data frame car ils offres unevue bidimentionnelle des données (comme un tableau excel), et c'est exactement ce que l'onva chercher à utiliser pour nos modèles.

3.3. Scikit-learn

une bibliothèque Python, fournit une large gamme d'algorithmes d'apprentissage supervisé et non supervisé. Elle s'appuie sur des technologies familières telles que NumPy, pandas et Matplotlib.

.

3.4. Matplotlib

est une bibliothèque complète permettant de créer des visualisations statiques, animéeset interactives en Python. Matplotlibrendles choses faciles et les choses difficiles possibles.

3.5. Os

Le module OS en python fournit des fonctions d'interaction avec le système d'exploitation. ‘OS’ fait partie des modules utilitaires standard de Python. Ce module fournit un moyen portable d'utiliser les fonctionnalités dépendant du système d'exploitation. Les modules ‘os’ et ‘os.path’ incluent de nombreuses fonctions pour interagir avec le système de fichiers.

3.6. PyTorch

C'est une bibliothèque open-source d'apprentissage automatique développée par Facebook, principalement utilisée pour le développement et la mise en œuvre de réseaux de neurones profonds.

4. Étapes de classification

Voici les étapes pour mettre en œuvre le système:

4.1. Importation des bibliothèques

Nous avons utilisé les libraires suivantes:

Figure1:Importer les bibliothèques nécessaires

```
#Importer les bibliothèques néccessaires.
import pandas as pd
import numpy as np
import os
from sklearn.preprocessing import LabelEncoder
import numpy as np
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import xgboost as xgb
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
from imblearn.over_sampling import SMOTES
```

4.2. Concaténation des tableaux

Nous avons utilisé la fonction pd.read_csv () de la bibliothèque pandas pour charger un fichierCSV dans un DataFrame. Dans notre cas, le fichier CSV que nous avons chargé Il est localisé dans le chemin spécifié.

```
#Concaténation des tableaux
d1 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Friday-WorkingHours-Morning.pcap_ISCX.csv')
d2 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv')
d3 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv')
d4 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Monday-WorkingHours.pcap_ISCX.csv')
d5 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX.csv')
d6 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv')
d7 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Tuesday-WorkingHours.pcap_ISCX.csv')
d8 = pd.read_csv('/kaggle/input/network-intrusion-dataset/Wednesday-workingHours.pcap_ISCX.csv')
df = pd.concat([d1,d2,d3,d4,d5,d6,d7,d8], axis=0)
```

Figure2:Chargement de données

4.3. Extraction caractéristiques et Labels

La dernière colonne du Data Frame df contient les étiquettes de données.

```
df[' Label'].value_counts().sum
```

```
<bound method NDFrame._add_numeric_operations.<locals>.sum of  Label
BENIGN                         2273097
DoS Hulk                        231073
PortScan                        158930
DDoS                            128027
DoS GoldenEye                    10293
FTP-Patator                       7938
SSH-Patator                       5897
DoS slowloris                     5796
DoS Slowhttptest                  5499
Bot                               1966
Web Attack ? Brute Force          1507
Web Attack ? XSS                   652
Infiltration                        36
Web Attack ? Sql Injection          21
Heartbleed                          11
Name: count, dtype: int64>
```

Figure 3:Extraction de la colonne de données cible du DataFrame

4.4. convertir le type de donnes

Notre type de dataset définie Object nous utilisant la conversion Object ver numérique.

4.5. Nettoyage des données

Nettoyage de données est l'étape la plus importante avant d'analyser ou modéliser des données.

```
#Nettoyage des données..
df = df.fillna(0)  # Replace NaN with 0
df = df.replace([np.inf, -np.inf], 0)
```

```
df.isnull().sum()
```

```
 Destination Port                0
 Flow Duration                   0
 Total Fwd Packets               0
 Total Backward Packets          0
Total Length of Fwd Packets      0
                                ..
Idle Mean                        0
 Idle Std                        0
 Idle Max                        0
 Idle Min                        0
 Label                           0
Length: 79, dtype: int64
```

Figure5:nettoyer les donnes

4.6. Normalisation des données

Cette étape est couramment utilisée dans le prétraitement des données pour améliorer les performances des modèles d'apprentissage automatique, en particulier pour les algorithmes sensibles à l'échelle des caractéristiques.

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Figure 6: normaliser des donnes

4.7 Étape de division

La bibliothèque scikit-learn(sklearn) en Python, qui est largement utilisée pour diviser un ensemblede données en ensembles d'entraînement et de test.

4.7. Entraînement du modèle

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X1, y1, test_size=0.3, random_state=42)
```

```
Epoch 1/50
11188/11188 [==============================] - 649s 58ms/step - loss: 650400.0625 - val_loss: 506165.
9062
Epoch 2/50
11188/11188 [==============================] - 643s 58ms/step - loss: 335628.8125 - val_loss: 271590.
4062
Epoch 3/50
11188/11188 [==============================] - 646s 58ms/step - loss: 146122.2969 - val_loss: 183092.
0625
Epoch 4/50
11188/11188 [==============================] - 645s 58ms/step - loss: 98399.8984 - val_loss: 196015.6
094
Epoch 5/50
11188/11188 [==============================] - 636s 57ms/step - loss: 78212.5156 - val_loss: 382857.4
062
Epoch 6/50
11188/11188 [==============================] - 637s 57ms/step - loss: 65739.3047 - val_loss: 187292.4
531
Epoch 7/50
11188/11188 [==============================] - 633s 57ms/step - loss: 58160.9805 - val_loss: 132306.7
656
Epoch 8/50
11188/11188 [==============================] - 633s 57ms/step - loss: 51849.0703 - val_loss: 183636.1
250
```

```
Epoch 9/50
11188/11188 [==============================] - 642s 57ms/step - loss: 47754.7148 - val_loss: 328592.4
375
Epoch 10/50
11188/11188 [==============================] - 641s 57ms/step - loss: 43978.0000 - val_loss: 282408.2
188
Epoch 11/50
11188/11188 [==============================] - 641s 57ms/step - loss: 40954.3945 - val_loss: 156910.3
594
Epoch 12/50
11188/11188 [==============================] - 641s 57ms/step - loss: 39494.0938 - val_loss: 136729.3
906
Epoch 13/50
11188/11188 [==============================] - 640s 57ms/step - loss: 36886.3945 - val_loss: 456797.7
500
Epoch 14/50
11188/11188 [==============================] - 640s 57ms/step - loss: 35581.3828 - val_loss: 152773.2
812
Epoch 15/50
11188/11188 [==============================] - 643s 57ms/step - loss: 34820.9180 - val_loss: 138943.8
438
Epoch 16/50
11188/11188 [==============================] - 644s 58ms/step - loss: 32925.9453 - val_loss: 154338.7
188
```

Figure10:Entraînement du modèle

5. Conclusion

Dans cette annexe, nous avons exploré en détail le développement de notre système, en fournissant une description complète des a mise en œuvre. Nous avons examiné les différentes composantes et étapes clés qui ont contribué à la réalisation du système.

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جامعة 20 أوت 1955- سكيكدة
كلية العلوم
مكتبة الكلية
Université 20 Août 1955 - Skikda
Faculté des Sciences
Bibliothèque de la Faculté

سكيكدة في / / 2024

بطاقة معلومات خاصة بمذكرة التخرج

اسم و لقب الطالب : رقم التسجيل :

Filali Berrezoug * 199422148 02 *

\ * \ *

\ * \ *

\ * \ *

اسم و لقب المشرف على المذكرة : Bougamouza Fateh

عنوان المذكرة : "Détection d'intrusions dans les Réseaux Informatique à l'aide de l'Apprentissage Automatique"

القسم : Informatique

المستوى : Master 2

التخصص : Réseaux et Systèmes Distribués (RSD)

الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

وزارة التعليـــــــم العالـــــــي و البحث العلمـــــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université 20 Août 1955- skikda-

Faculté des Sciences

Département d'Informatique

جـامعة 20 أوت 1955- سكيكدة

كـــــلية العلـــــوم

قسم الاعلام الالي

الرقم : ..../ ق/ ا/ أ/ ت.ع/2024

Autorisation de Dépôt de Mémoire de Master

Je soussigné: .....Bouganouza Fateh.....

Certifie que l'étudiant(e) :..Filali Bennzouag.....

Spécialité : ...Réseaux et systèmes distribués.....

Ayant soutenu le projet intitulé :..Détection d'intrusions...
.à l'aide de l'apprentissage automatique.....
.....

A apporté les corrections nécessaires sur son manuscrit de Master

جامعة 20 أوت 1955 - سكيكدة
قسم الإعلام الآلي
كلية العلوم

Signature de l'encadreur