الجمهورية الجزائرية الديمقراطية الشعبية
**République Algérienne Démocratique et Populaire**

وزارة التعليم العالي والبحث العلمي
**Ministère de l'Enseignement Supérieur et de la Recherche Scientifique**

جامعة سعد دحلب البليدة
**Université SAAD DAHLAB de BLIDA**

كلية التكنولوجيا
**Faculté de Technologie**

قسم الإلكترونيك
**Département d'Électronique**

# Mémoire de Projet de Fin d'Études

présenté par

LIZLI RAFIK

&

KECIOUR ANES

pour l'obtention du diplôme de Master 2 en télécommunication option RT

Thème

# Détection d'intrusions réseaux en utilisant le Machine Learning

Proposé par : Y. Kabir

Année Universitaire 2022-2023

# Remerciements

*Nous souhaitons exprimer notre profonde gratitude et nos sincères remerciements à tous ceux qui ont contribué de près ou de loin au bon déroulement de notre mémoire de fin d'études et à l'élaboration de ce travail.*

*En premier lieu, nous tenons à exprimer notre reconnaissance à notre encadreur, M. Y. Kabir, pour son soutien et ses conseils inestimables qui ont été essentiels pour mener à bien notre travail. Sa disponibilité et son expertise ont grandement enrichi notre parcours. Nous lui sommes sincèrement reconnaissants.*

*Nous souhaitons également adresser nos vifs remerciements au membre du jury prestigieux ainsi qu'à tout le personnel du département de Génie Électronique de l'Université de SAAD DAHLAB Blida 1. Nous sommes particulièrement reconnaissants envers nos professeurs qui nous ont accompagnés tout au long de ces cinq années. Leur enseignement, leurs encouragements et leurs remarques constructives ont joué un rôle déterminant dans notre réussite académique.*

*Nous souhaitons enfin exprimer notre chaleureux remerciement à nos chers parents, dont le soutien inconditionnel et la patience ont été d'une importance capitale. Leur présence et leur encouragement dans les moments difficiles ont été une source de motivation inestimable.*

*Nous tenons à exprimer notre gratitude envers toutes les personnes qui ont contribué, par leurs conseils et leurs compétences, à la réalisation de ce mémoire, même si nous ne pouvons les citer tous individuellement. Votre implication et votre soutien ont joué un rôle clé dans la concrétisation de ce projet.*

*Nous sommes profondément reconnaissants pour tout ce que vous avez fait et pour le temps précieux que vous nous avez consacré.*

**ملخص:** : نشهد حاليًا زيادة مستمرة في التهديدات المتعلقة بالاختراقات الشبكية. في هذه الدراسة، نركز على اكتشاف الاختراقات في الشبكة باستخدام تقنيات التعلم الآلي. هدفنا هو استكشاف تقنيات الكشف المختلفة من خلال الاستفادة من نماذج التعلم الآلي. يتم تقييم النتائج التي نحصل عليها بعناية ومقارنتها لتحديد النماذج الأكثر فعالية. نسلط الضوء على أهمية التعلم الآلي في اكتشاف الاختراقات، حيث يتيح الكشف المبكر والدقيق للاختراقات الشبكية. بالإضافة إلى ذلك، نحدد الفرص المتاحة للبحث المستقبلي لتعزيز تقنيات منع الاختراق والاستجابة، مما يضمن في النهاية تعزيز أمان الشبكة .

وفي الختام، يُوصَى باستخدام XGBoost لاكتشاف الاختراقات في الشبكة بسبب دقته العالية وقدرته على تقليل أخطاء التصنيف وأدائه الجيد من حيث الاستدعاء.

**الكلمات الرئيسية**: الاختراقات الشبكية، اكتشاف الاختراقات في الشبكة، التعلم الآلي، نماذج التعلم الآلي، الأمان، XGBoost

**Résumé :** Nous assistons de nos jours à une augmentation constante des menaces liées aux intrusions réseau. Dans le cadre de cette étude, nous nous concentrons sur la détection d'intrusions réseau en utilisant le Machine Learning. Notre objectif est d'explorer différentes techniques de détection en exploitant les modèles de Machine Learning. Les résultats que nous obtenons sont soigneusement évalués et comparés afin de déterminer les modèles les plus performants. Nous mettons en évidence l'importance du Machine Learning dans la détection d'intrusions, car il permet une détection précoce et précise des intrusions réseau. De plus, nous identifions des opportunités d'amélioration pour les recherches futures, dans le but de renforcer les techniques de prévention et de réponse aux intrusions, et ainsi garantir une sécurité accrue des réseaux.

En conclusion, XGBoost est recommandé pour la détection d'intrusions réseau en raison de sa précision élevée, de sa capacité à minimiser les erreurs de classification et de sa bonne performance en termes de rappel.

**Mots clés :** intrusions réseau, détection d'intrusions réseau, Machine Learning, Les modèles de Machine Learning, sécurité, XGBoost

**Abstract:** We are currently witnessing a constant increase in threats related to network intrusions. In this study, we focus on network intrusion detection using Machine Learning. Our objective is to explore different detection techniques by leveraging Machine Learning models. The results we obtain are carefully evaluated and compared to determine the most effective models. We highlight the importance of Machine Learning in intrusion detection, as it enables early and accurate detection of network intrusions. Additionally, we identify opportunities for future research to enhance intrusion prevention and response techniques, ultimately ensuring enhanced network security.

In conclusion, XGBoost is recommended for network intrusion detection due to its high accuracy, ability to minimize classification errors, and good performance in terms of recall.

**Keywords:** network intrusions, network intrusion detection, Machine Learning, Machine Learning models, security, XGboost

## Listes des acronymes et abréviations :

| | | |
|---|---|---|
| AIDS | : | Anomaly-Based Intrusion Detection System |
| ANN | : | Artificial Neural Network |
| APIDS | : | Application Protocol-Based Intrusion Detection System |
| CNN | : | Convolutional Neural Network |
| CPUs | : | Central Processing Units |
| DL | : | Deep Learning |
| DoS | : | Denial of Service |
| DT | : | Decision Tree |
| FN | : | False Negative |
| FP | : | False Positive |
| HIDS | : | Host-Based Intrusion Detection System |
| HTTP | : | Hypertext Transfer Protocol |
| HTTPS | : | Hypertext Transfer Protocol Secure |
| IA | : | Intelligence Artificielle |
| ICMP | : | Internet Control Message Protocol |
| IDS | : | Intrusion Detection System |
| IP | : | Internet Protocol |
| KDD | : | Knowledge Discovery in Databases |
| KNN | : | K-Nearest Neighbors |
| LR | : | Logistic Regression |
| LSTM | : | Long Short-Term Memory |
| ML | : | Machine Learning |
| MTTR | : | Mean Time To Repair |
| NB | : | Naive Bayes |
| NIDS | : | Network Intrusion Detection System |
| NNIDS | : | Network-Based Intrusion Detection System |
| NSL | : | Network Security Laboratory |
| PCA | : | Principal Component Analysis |
| PIDS | : | Protocol-Based Intrusion Detection System |
| R2L | : | Right-to-Left |
| REJ | : | Reject |
| RF | : | Random Forest |
| RL | : | Reinforcement Learning |
| RNN | : | Recurrent Neural Network |

| | | |
|---|---|---|
| SF | : | Security Feature |
| SIDS | : | Signature-Based Intrusion Detection Systems |
| SVM | : | Support Vector Machine |
| TCP | : | Transmission Control Protocol |
| TN | : | True Negative |
| TP | : | True Positive |
| U2R | : | User-to-Root |
| UDP | : | User Datagram Protocol |
| XGBoost | : | eXtreme Gradient Boosting |

# Table des matières

# Liste des figures

# Liste des tableaux

# Introduction générale

# Introduction générale

La sécurité des réseaux est devenue une préoccupation majeure dans le domaine de l'informatique et des communications, notamment en raison de l'augmentation constante des cyberattaques et des menaces qui pèsent sur les systèmes informatiques. La détection d'intrusions joue un rôle essentiel dans la préservation de l'intégrité des réseaux en identifiant et en prévenant les activités malveillantes qui pourraient compromettre la confidentialité, l'intégrité et la disponibilité des données [1]. Cependant, les techniques traditionnelles de détection d'intrusions basées sur des règles et des signatures se révèlent souvent insuffisantes pour faire face à la sophistication croissante des attaques. Les attaquants utilisent des méthodes de plus en plus évoluées pour contourner ces systèmes de détection classiques, rendant impératif le développement de nouvelles approches plus efficaces.

C'est dans ce contexte que le machine learning et le deep learning ont suscité un intérêt croissant en tant qu'outil prometteur pour améliorer la détection d'intrusions réseau. Le machine learning permet d'exploiter de grandes quantités de données et d'identifier des schémas, des comportements anormaux et des signatures non détectables par les méthodes traditionnelles. En utilisant des techniques de deep learning et de machine learning, telles que les réseaux de neurones profonds, il devient possible de capturer des caractéristiques complexes et de détecter des attaques insidieuses avec une plus grande précision [2].

L'objectif de ce projet de fin d'études est d'explorer l'utilisation du machine learning, en mettant l'accent sur les techniques de deep learning, pour la détection d'intrusions réseau. Nous nous pencherons sur les concepts de base de la détection d'intrusions, les techniques classiques utilisées jusqu'à présent et les principes fondamentaux du machine learning appliqués à ce domaine spécifique. Une revue de la littérature permettra d'explorer les travaux existants dans le domaine de la détection d'intrusions en utilisant des approches basées sur le machine learning, offrant ainsi une compréhension approfondie des avancées récentes, des méthodologies utilisées et des résultats obtenus. Cette revue de la littérature nous permettra également d'identifier les lacunes actuelles et les opportunités de recherche dans le domaine [3].

La méthodologie adoptée pour ce projet comprendra l'utilisation d'outils de développement tels que Python et Jupyter Notebook, ainsi que des ensembles de données largement utilisés, tels que le jeu de données NSL-KDD, qui fournit une représentation réaliste du trafic réseau et est couramment utilisé dans la communauté de la détection d'intrusions. Nous aborderons également les étapes de prétraitement des données, qui comprennent le nettoyage, la

normalisation et la sélection des caractéristiques pertinentes pour améliorer la qualité des données d'entrée pour les modèles de machine learning. Différents modèles de machine learning, tels que KNN, SVM, Decision tree, Random Forest, CNN, RNN, régression logistique, xgboost et naive bayes, seront explorés et évalués pour leur efficacité dans la détection d'intrusions réseau.

Dans la section des résultats et analyses, nous présenterons les résultats obtenus en utilisant ces modèles de machine learning sur les ensembles de données de détection d'intrusions. Nous évaluerons les performances des modèles en termes de précision, de rappel, de taux de faux positifs et d'autres mesures de performance appropriées. De plus, nous comparerons les résultats des différents modèles afin de déterminer celui qui présente les meilleures performances dans notre contexte spécifique.

En conclusion, nous synthétiserons les résultats et les contributions de ce travail, en mettant en évidence les avantages et les limites des approches de détection d'intrusions basées sur le machine learning. Nous discuterons également des perspectives pour des travaux futurs, telles que l'amélioration des modèles existants, l'exploration de nouvelles techniques de machine learning spécifiques à la détection d'intrusions et l'intégration de l'intelligence artificielle pour une détection encore plus avancée des intrusions réseau.

# Chapitre I
# État de l'art

# Chapitre I : État de l'art

## 1.1 Introduction

La détection d'intrusion est une technique de sécurité importante pour surveiller les activités du réseau et détecter les activités suspectes. Les techniques classiques de détection d'intrusion sont limitées, car elles ne peuvent pas détecter les nouvelles attaques ou les attaques modifiées. Le Machine Learning offre une nouvelle approche en utilisant des algorithmes d'apprentissage pour détecter les anomalies dans les données du réseau. Dans ce chapitre, nous passerons en revue les concepts de base, les techniques classiques et les principes du Machine Learning appliqués à la détection d'intrusion, ainsi que les travaux de recherche récents dans le domaine. Nous examinerons les avantages et les limites de chaque technique, ainsi que les défis et les opportunités dans la mise en œuvre de la détection d'intrusion basée sur le Machine Learning. L'utilisation du Machine Learning peut améliorer l'efficacité de la détection d'intrusion et renforcer la sécurité des entreprises.

## 1.2 Concepts de base en détection d'intrusion

En général, la conception de base de la détection d'intrusion implique la surveillance constante du trafic réseau à la recherche de comportements suspects, la comparaison de ces comportements avec des règles prédéfinies ou des modèles appris, et la génération d'alertes en cas de détection d'une activité malveillante ou suspecte.

### 1.2.1 Systèmes de Détection d'Intrusion (IDS)

Le concept de surveillance de l'activité des utilisateurs à travers les journaux et les enregistrements informatiques a été introduit pour la première fois en 1980 par Jim Anderson. Cette pratique visait à protéger les informations contre l'accès par des utilisateurs internes ou externes non autorisés, ainsi que contre les "misfeasors", les utilisateurs qui ont mal utilisé leurs privilèges. C'était le début des IDS basés sur l'hôte.

Depuis lors, l'idée a été étudiée et développée pour suivre l'utilisation publique croissante d'Internet et ses vulnérabilités. Le premier système de détection d'intrusion en temps réel s'appelle Intrusion Detection Expert System (IDES) ; un produit d'une recherche menée au Computer Science Laboratory de SRI International. IDES est un système de détection d'intrusion en temps réel indépendant qui combine à la fois la détection basée sur des anomalies et basée sur des règles ; la détection basée sur des anomalies utilise des algorithmes statistiques tandis que le système expert a été utilisé pour construire les composants basés sur des règles. On dit qu'il est indépendant, car il n'est pas lié à un système particulier, à un environnement, à une vulnérabilité ou à un type d'intrusion particulier. Le modèle IDES lancés en 1986 était à usage général et le cadre pouvait être utilisé comme fondation pour développer un IDS beaucoup plus robuste et puissant.

La technologie des IDS est devenue un domaine très populaire et bien étudié, en raison de sa demande dans l'industrie. L'intérêt incessant pour cette technologie a continuellement amélioré la performance et la précision des IDS.

Depuis la formation d'IDES, il y a beaucoup plus de produits de détection d'intrusion sur le marché pour que les utilisateurs puissent choisir et mettre en œuvre. En 1999, Kevin Richards [4] a effectué une revue de cinq produits IDS différents pour évaluer leur performance dans un environnement de production. Dans sa revue, il a souligné l'importance du moteur de traitement de paquets dans les IDS. Si le moteur n'est pas efficace et efficient, le capteur ou le IDS commencera à perdre des paquets et cela réduira la capacité à détecter des attaques, surtout s'il s'agit d'une attaque avec plusieurs paquets qui doivent être assemblés.

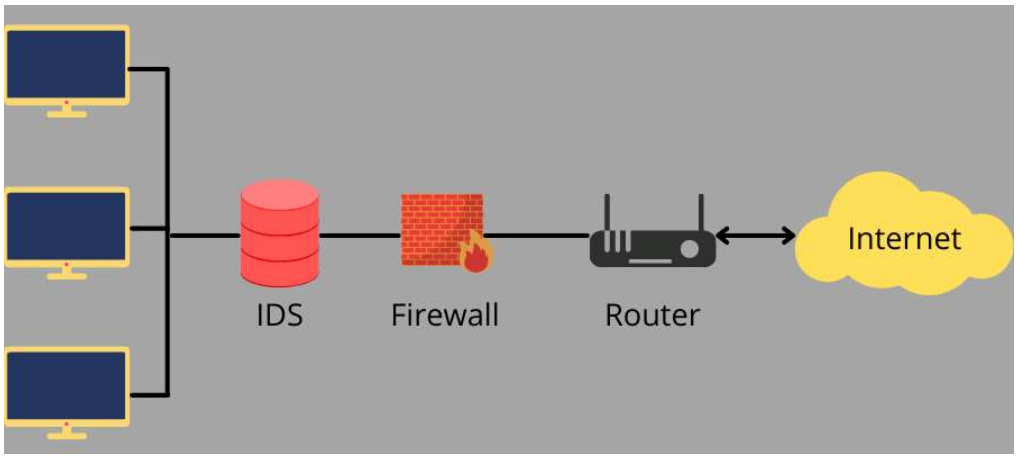


Figure 1. 1 : Architecture de sécurité incluant un système de détection d'intrusion (IDS) [4].

### 1.2.2 Types d'intrusions et de menaces

Il existe quatre grandes catégories d'attaques en réseau. Chaque attaque sur un réseau peut être classée dans l'une de ces catégories.

**a. Déni de service (DoS)**

Une attaque DoS est un type d'attaque dans laquelle le pirate rend les ressources de mémoire trop occupées pour répondre aux demandes de réseau légitimes, empêchant ainsi les utilisateurs d'accéder à une machine, par exemple avec des attaques telles que Apache, Smurf, Neptune, ping of death, back, mail bomb, tempête UDP, etc.

**b. Attaques à distance sur un utilisateur (R2L)**

Une attaque à distance sur un utilisateur est une attaque dans laquelle un utilisateur envoie des paquets à une machine sur Internet à laquelle il n'a pas accès, dans le but d'exposer les vulnérabilités de la machine et d'exploiter les privilèges qu'un utilisateur local aurait sur l'ordinateur, par exemple avec des attaques

telles que xlock, guest, xnsnoop, phf, sendmail, dictionnaire, etc.

**c. Attaques d'utilisateur à super-utilisateur (U2R)**

Ces attaques sont des exploitations dans lesquelles le pirate commence sur le système avec un compte utilisateur normal et tente d'exploiter les vulnérabilités pour obtenir des privilèges de super utilisateur, par exemple avec des attaques telles que Perl, Xterm.

**d. Sonde**

Une sonde est une attaque dans laquelle le pirate analyse une machine ou un périphérique de réseau afin de déterminer des faiblesses ou des vulnérabilités qui pourront être exploitées ultérieurement pour compromettre le système. Cette technique est couramment utilisée dans l'exploration de données, par exemple avec des attaques telles que satan, saint, portsweep, mscan, nmap, etc.

Deux types d'attaques différents ont été inclus pour cette étude : le SYN Flood (Neptune) et Satan. Ces deux types d'attaques ont été sélectionnés à partir de deux catégories d'attaques différentes (déni de service et sondes) pour vérifier la capacité du système de détection d'intrusion à identifier des attaques provenant de différentes catégories.

**e. Le SYN Flood (Neptune)**

Le SYN Flood est une attaque de déni de service à laquelle chaque implémentation TCP/IP est vulnérable (dans une certaine mesure). Pour distinguer une attaque Neptune, le trafic réseau est surveillé pour un certain nombre de paquets SYN simultanés destinés à une machine particulière.

**f. Satan**

Satan est une intrusion de sondage, qui analyse automatiquement un réseau d'ordinateurs pour recueillir des informations ou trouver des vulnérabilités connues[5].

### 1.2.3 Catégories de détection d'intrusion

Bien que tous les systèmes de détection d'intrusion remplissent le même objectif, ils fonctionnent de manière légèrement différente. Il existe cinq types de IDS au total. Examinons les détails, avantages et inconvénients de chacun.

**a. Système de détection d'intrusion réseau**

Un système de détection d'intrusion réseau (NIDS) surveille tout le trafic qui passe par votre réseau grâce à un ou plusieurs points de contact. Pour utiliser un NIDS, il doit être installé sur un matériel de votre infrastructure réseau. Une fois en place, il échantillonnera chaque paquet de données qui circule à travers

lui.

Cependant, tous les paquets ne nécessitent pas une analyse, car cela risque de manquer une tentative d'intrusion en raison d'une surcharge d'informations. Pour remédier à ce problème, il est possible de créer un ensemble de règles qui définissent les types de paquets que votre NIDS détectera et stockera.

Les avantages du NIDS sont les suivants :

- Il peut analyser tout le trafic entrant et sortant.
- Il détecte les événements en temps réel, permettant des temps de réponse rapides.
- Il est plus difficile à détecter par les intrus.
- Il peut être placé de manière stratégique dans des zones critiques.

Cependant, les NIDS ne sont pas parfaits. Les inconvénients potentiels sont la maintenance pratique, car il nécessite souvent une interaction manuelle avec lui, et une faible spécificité. Plus un NIDS analyse de trafic, plus il risque de manquer de spécificité et de ne pas détecter les signes d'une intrusion.

**b. Système de détection d'intrusion de nœud de réseau :**

Un système de détection d'intrusion de nœud de réseau (NNIDS) est techniquement une variation d'un NIDS, mais comme il fonctionne différemment, nous le considérerons comme un type différent de IDS. Un NNIDS analyse également les paquets qui passent à travers lui. Cependant, au lieu de s'appuyer sur un dispositif central pour surveiller l'ensemble du trafic réseau, le système surveille chaque nœud connecté à votre réseau.

Cette différenciation comporte plusieurs avantages, tels que :

- Des vitesses plus élevées - étant donné que la quantité de trafic analysée par chaque agent NNIDS est réduite, le système peut fonctionner plus rapidement.
- Moins de ressources utilisées De la même manière, NNIDS utilise moins de ressources système. Ainsi, Nous pouvons facilement l'installer sur vos serveurs actuels.

Le principal inconvénient de l'option NNIDS est le besoin de plusieurs installations. Alors qu'un NIDS ne nécessite qu'un seul dispositif, NNIDS en nécessite plusieurs (un pour chaque serveur)

### c. Système de détection d'intrusion sur hôte :

Un système de détection d'intrusion sur hôte (HIDS) pousse l'indépendance de l'appareil d'un système de détection d'intrusion basé sur le réseau (NNIDS) un cran plus loin. Avec un HIDS, vous pouvez installer un logiciel IDS sur chaque appareil connecté à votre réseau. Les HIDS fonctionnent en prenant des "instantanés" de l'appareil qui leur est assigné. En comparant le dernier instantané aux enregistrements passés, le HIDS peut identifier les différences qui pourraient indiquer une intrusion.

Les HIDS présentent des avantages, car :

- Ils peuvent être installés sur des ordinateurs ou des serveurs
- Ils peuvent identifier l'appareil affecté
- Ils notifient les administrateurs si des fichiers système analytiques ont été modifiés ou supprimés
- Ils sont particulièrement efficaces contre les menaces internes

Malheureusement, les solutions HIDS peuvent souffrir d'une surveillance "après-coup". Parce que de nombreuses solutions HIDS s'appuient sur des journaux qui enregistrent les intrusions, votre temps moyen de réponse (MTTR) peut être plus lent dans l'ensemble. Par conséquent, l'utilisation appropriée d'un HIDS nécessite une surveillance fréquente.

### d. Système de détection d'intrusion basé sur le protocole

Un système de détection d'intrusion basé sur le protocole (PIDS) est un IDS spécifique qui surveille le protocole en cours d'utilisation. Dans la pratique, ce système analyse généralement le flux de protocole HTTP ou HTTPS entre vos appareils et le serveur. Dans la plupart des cas, un PIDS sera placé à l'avant d'un serveur. Le système peut protéger votre serveur web en surveillant le trafic entrant et sortant. Étant donné qu'ils se concentrent sur le protocole (la façon dont les appareils transmettent des informations au sein d'un réseau), les PIDS ne sont pas nécessairement une solution IDS complète. Cependant, ils peuvent compléter une solution de cybersécurité déjà robuste.

### e. Système de détection d'intrusion basé sur le protocole de l'application

Un système de détection d'intrusion basé sur le protocole de l'application (APIDS) est un type d'IDS qui se spécialise dans la sécurité des applications logicielles. Généralement associés aux systèmes de détection d'intrusion sur hôte (HIDS), les APIDS surveillent les communications qui ont lieu entre les applications et le serveur. Un APIDS est généralement installé sur des groupes de serveurs.
Comme pour un PIDS, un APIDS ne résoudra probablement pas tous vos besoins de surveillance réseau.

Cependant, il peut compléter d'autres types d'IDS.

## 1.3 Techniques classiques de détection d'intrusion

Selon le type de système de détection d'intrusion que vous choisissez, votre solution de sécurité reposera sur quelques méthodes de détection différentes pour vous protéger. Voici un bref aperçu de chacune d'entre elles.

### 1.3.1 Détection d'intrusion basée sur les signatures

Les systèmes de détection d'intrusion basés sur les signatures (SIDS) visent à identifier des modèles et à les faire correspondre à des signes connus d'intrusions. Un SIDS repose sur une base de données d'intrusions précédentes. Si l'activité au sein de votre réseau correspond à la "signature" d'une attaque ou d'une violation de la base de données, le système de détection notifie votre administrateur. Étant donné que la base de données est la colonne vertébrale d'une solution SIDS, des mises à jour fréquentes de la base de données sont essentielles, car les SIDS ne peuvent identifier que les attaques qu'ils reconnaissent. Par conséquent, si votre organisation devient la cible d'une technique d'intrusion jamais vue auparavant, aucune mise à jour de la base de données ne pourra vous protéger.

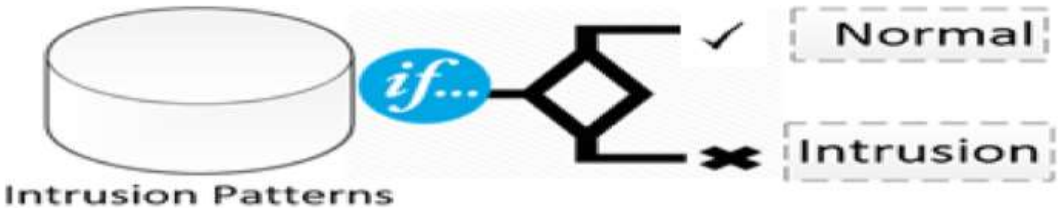


Figure 1. 2 : Le fonctionnement conceptuel des approches SIDS [6].

### 1.3.2 Détection d'intrusion basée sur les anomalies

En revanche, un système de détection d'intrusion basé sur les anomalies (AIDS) peut identifier ces nouvelles intrusions zero-day. Un SIDS utilise l'apprentissage automatique (ML) et les données statistiques pour créer un modèle de comportement "normal". Chaque fois que le trafic s'écarte de ce comportement typique, le système le signale comme suspect. Le principal problème d'AIDS par rapport à SIDS est le risque de faux positifs. Après tout, tous les changements ne sont pas le résultat d'activités malveillantes ; certains sont simplement des indications de changements dans le comportement organisationnel. Mais parce qu'un SIDS n'a pas de base de données d'attaques connues à référencer, il peut signaler toutes les anomalies comme des intrusions.

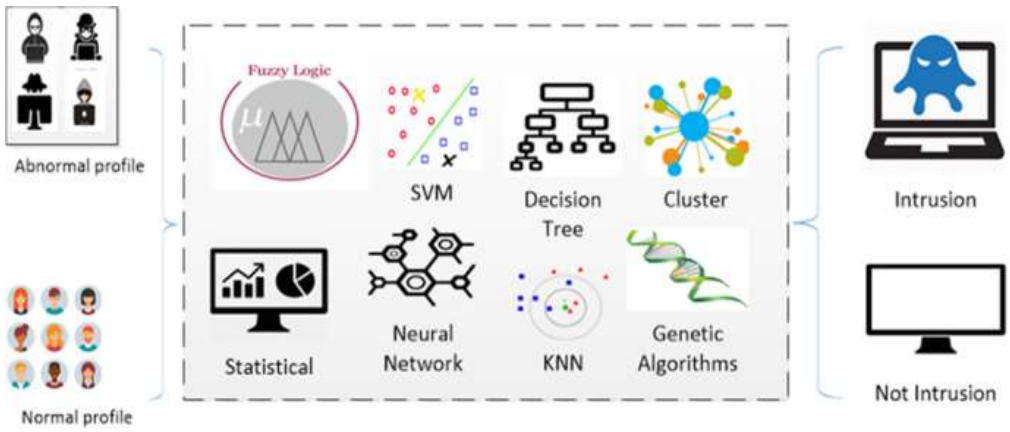


Figure 1. 3 : Le fonctionnement conceptuel des approches AIDS basées sur l'apprentissage automatique [6].

### 1.3.3 Détection d'intrusion hybride

Un système hybride combine le meilleur des deux mondes. En examinant les modèles et les événements ponctuels, un système de détection d'intrusion hybride peut signaler de nouvelles stratégies d'intrusion existantes et nouvelles. Le seul inconvénient d'un système hybride est l'augmentation encore plus grande des problèmes signalés. Cependant, étant donné que le but d'un IDS est de signaler les intrusions potentielles, il est difficile de considérer cette augmentation de signalements comme négative. [6]

## 1.4 Principes du machine Learning appliqués à la détection d'intrusion

### 1.4.1 Introduction au machine Learning

Le Machine Learning est une branche de l'Intelligence Artificielle qui permet aux ordinateurs d'apprendre automatiquement à partir de données d'entraînement et de s'améliorer au fil du temps sans être explicitement programmés. Les algorithmes de Machine Learning peuvent détecter des modèles dans les données et apprendre à partir d'eux pour faire leurs propres prédictions. Contrairement à la programmation traditionnelle, qui implique des instructions basées sur des structures SI-ALORS, le Machine Learning est un processus automatisé qui permet aux machines de résoudre des problèmes avec peu ou pas d'entrée humaine, et de prendre des décisions basées sur des observations passées. Bien que souvent utilisées de manière interchangeable, l'Intelligence Artificielle et le Machine Learning sont deux concepts différents, le Machine Learning étant un sous-ensemble de l'IA qui permet aux systèmes intelligents d'apprendre de manière autonome à partir des données. Le Machine Learning peut être utilisé pour travailler sur de grandes quantités de données et peut être beaucoup plus précis que les humains, ce qui permet de gagner du temps et de l'argent sur des tâches et des analyses telles que la résolution de problèmes clients, l'automatisation des tickets de support et l'extraction de données à partir de sources internes et sur Internet.

De nos jours, l'utilisation de l'apprentissage automatique (Machine Learning) est en augmentation dans toutes les industries. L'apprentissage automatique est une branche de l'informatique qui permet à un ordinateur d'apprendre sans être programmé explicitement, comme l'a mentionné Arthur Samuel en 1959. L'apprentissage automatique donne une explication brève des algorithmes à partir desquels nous pouvons faire des prédictions sur les données. Les systèmes d'apprentissage automatique apprennent donc des programmes à partir des données existantes.

### 1.4.2 Types d'Algorithmes Machine Learning

Les algorithmes d'apprentissage automatique sont classés en apprentissage supervisé, apprentissage non supervisé et apprentissage par renforcement.

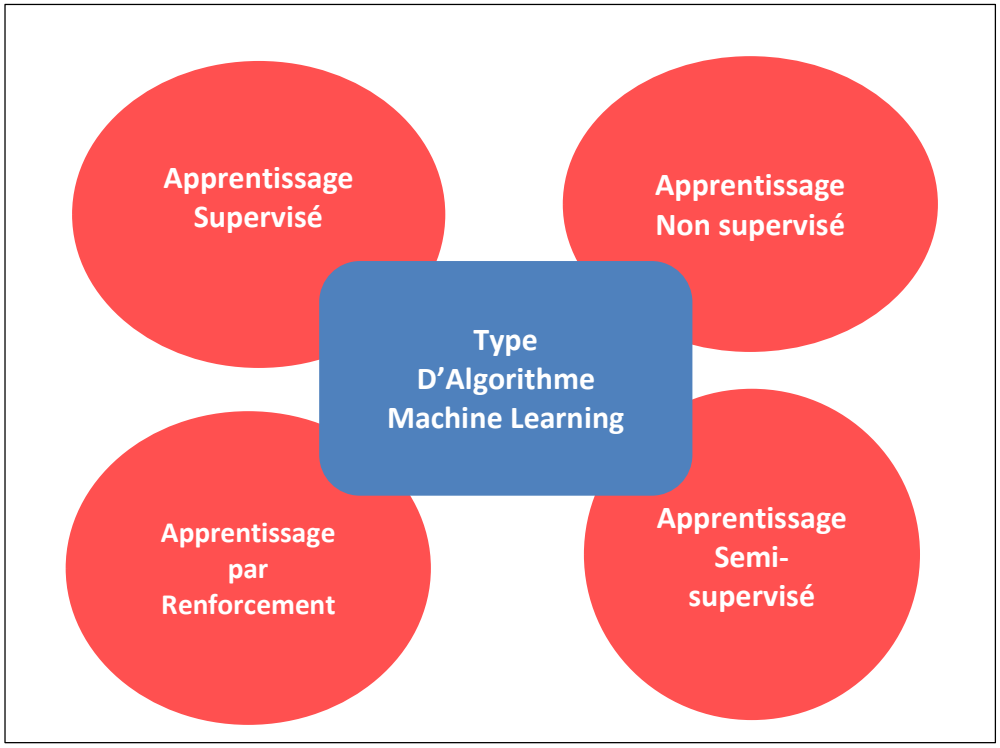


Figure 1. 4 : Classification de l'apprentissage automatique

#### A. Apprentissage supervisé

Les algorithmes et modèles d'apprentissage supervisé effectuent des prédictions en se basant sur des données d'entraînement étiquetées. Chaque échantillon d'entraînement comprend une entrée et une sortie désirée. Un algorithme d'apprentissage supervisé analyse ces échantillons et établit une inférence - une conjecture éduquée lors de la détermination des étiquettes pour des données non vues. C'est l'approche la plus courante et populaire de l'apprentissage automatique. C'est "supervisé", car ces modèles doivent être nourris avec des

données d'échantillonnage étiquetées pour apprendre. Les données sont étiquetées pour indiquer à la machine les motifs (mots et images similaires, catégories de données, etc.) qu'elle doit rechercher et reconnaître les connexions

**a. Classification en apprentissage automatique supervisé.**

Il existe plusieurs algorithmes de classification utilisés en apprentissage supervisé, parmi lesquels les machines à vecteurs de support (SVM) et les classificateurs bayésiens naïfs sont les plus courants. Dans les tâches de classification, la valeur de sortie est une catégorie avec un nombre fini d'options.

**b. La régression en apprentissage supervisé en Machine learning**

En apprentissage supervisé, la régression consiste à prédire une valeur numérique continue, telle que la probabilité qu'un événement se produise, qui peut avoir n'importe quelle valeur numérique dans une certaine plage.

## B. Apprentissage non supervisé

Les algorithmes d'apprentissage non supervisé découvrent des informations et des relations dans des données non étiquetées. Dans ce cas, les modèles sont alimentés en données d'entrée, mais les résultats désirés sont inconnus, ils doivent donc tirer des conclusions à partir d'indices circonstanciels, sans aucune guidance ou formation. Les modèles ne sont pas entraînés avec la « bonne réponse », ils doivent donc trouver des motifs par eux-mêmes. L'un des types les plus courants d'apprentissages non supervisés est le clustering, qui consiste à regrouper des données similaires. Cette méthode est principalement utilisée pour l'analyse exploratoire et peut aider à détecter des modèles ou des tendances cachées. Par exemple, l'équipe marketing d'une entreprise de commerce électronique pourrait utiliser le clustering pour améliorer la segmentation des clients. Avec un ensemble de données de revenus et de dépenses, un modèle d'apprentissage automatique peut identifier des groupes de clients ayant des comportements similaires. La segmentation permet aux marketeurs d'adapter les stratégies pour chaque marché clé. Ils pourraient offrir des promotions et des remises aux clients à faible revenu qui dépensent beaucoup sur le site, afin de récompenser la fidélité et d'améliorer la rétention.

## C. Apprentissage par renforcement

L'apprentissage par renforcement (RL) concerne la façon dont un agent logiciel (ou un programme informatique) devrait agir dans une situation pour maximiser la récompense. En bref, les modèles d'apprentissage renforcé tentent de déterminer le meilleur chemin possible à suivre dans une situation donnée. Ils le font par essais et erreurs. Comme il n'y a pas de données d'entraînement, les machines apprennent de leurs propres erreurs et choisissent les actions qui mènent à la meilleure solution ou à la

récompense maximale.

Cette méthode d'apprentissage automatique est principalement utilisée dans la robotique et les jeux. Les jeux vidéo démontrent une relation claire entre les actions et les résultats, et peuvent mesurer le succès en gardant une note. Par conséquent, ils constituent un excellent moyen d'améliorer les algorithmes d'apprentissage par renforcement.

## D. Apprentissage semi-supervisé

Dans l'apprentissage semi-supervisé, les données d'entraînement sont divisées en deux. Une petite quantité de données étiquetées et un ensemble plus important de données non étiquetées. Dans ce cas, le modèle utilise les données étiquetées comme entrée pour faire des inférences sur les données non étiquetées, fournissant des résultats plus précis que les modèles d'apprentissage supervisé réguliers. Cette approche gagne en popularité, en particulier pour les tâches impliquant de grands ensembles de données tels que la classification d'images. L'apprentissage semi-supervisé ne nécessite pas un grand nombre de données étiquetées, il est donc plus rapide à mettre en place, plus rentable que les méthodes d'apprentissage supervisé et idéal pour les entreprises qui reçoivent d'énormes quantités de données. Le diagramme suivant montre la classification des algorithmes d'apprentissage automatique. Les algorithmes de machine learning (ML) sont utilisés pour résoudre les problèmes de détection d'intrusion (IDS) en se basant sur la conception de classificateur unique, de classificateur hybride et de classificateur d'ensemble. Nous allons discuter en détail de ces trois classificateurs et de leurs algorithmes.

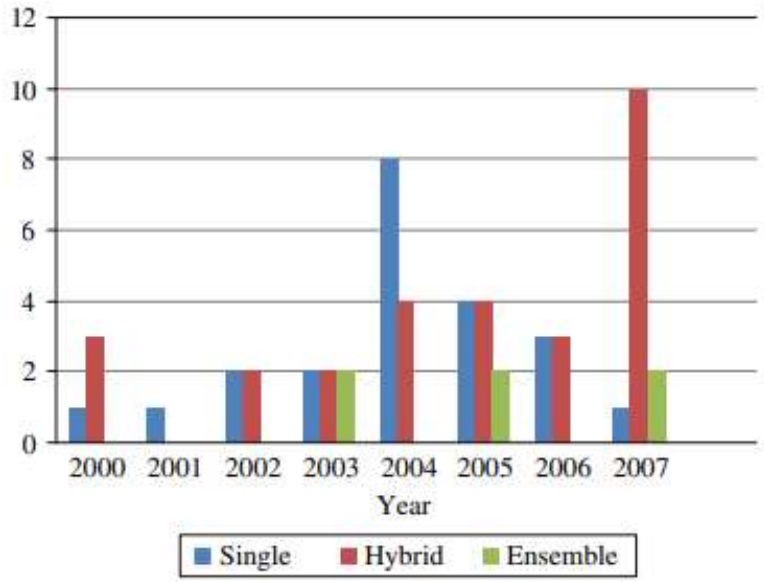


Figure 1. 5 : Répartition annuelle des articles pour les types de conception de classificateurs [7].

### 1.4.3 Classificateurs

Les classificateurs sont des algorithmes d'apprentissage automatique (machine learning) qui sont utilisés pour classer des données dans des catégories spécifiques en fonction de leurs caractéristiques. Les classificateurs sont utilisés dans de nombreux domaines, tels que la reconnaissance d'images, la reconnaissance vocale, l'analyse de texte, la détection de spam et la prédiction des résultats financiers.

#### A. Classificateurs uniques

Un IDS développé par un algorithme d'apprentissage machine simple est connu sous le nom de classificateur unique ou classificateur simple. Les techniques d'apprentissage machine suivantes sont souvent utilisées comme classificateurs uniques.

#### a) Arbre de décision (Decision tree)

Un arbre de décision classe une étiquette inconnue à partir d'une liste de décisions. L'arbre de décision est très simple et facile à mettre en œuvre. Il est souvent utilisé comme classificateur unique. Un arbre de décision est divisé en arbre de régression et arbre de classification.

#### b) Naive Bayes

Naive Bayes montre de meilleures performances en classification en raison des relations simples. Il ne réalise qu'un seul balayage des données et la classification est donc facile. Naive Bayes effectue une probabilité conditionnelle en fonction de l'étiquette de classe donnée.

#### c) Random Forest

Le Random Forest est un algorithme d'apprentissage supervisé qui crée un ensemble de modèles d'arbres de décision et agrège leurs prédictions pour améliorer la précision globale et la stabilité du modèle. Chaque arbre de décision est construit à partir d'un sous-ensemble aléatoire des données d'entraînement et d'un sous-ensemble aléatoire des variables explicatives, ce qui permet de réduire l'overfitting et d'augmenter la généralisation du modèle.

#### d) K-plus proches voisins (KNN)

KNN est l'approche la plus simple pour la classification d'échantillons. Ici, différentes mesures de distance sont utilisées pour classer les échantillons. Le K-plus proches voisins trouve le nombre d'échantillons à partir des données d'entraînement qui sont proches de l'échantillon de test et attribue l'étiquette de classe la plus fréquente.

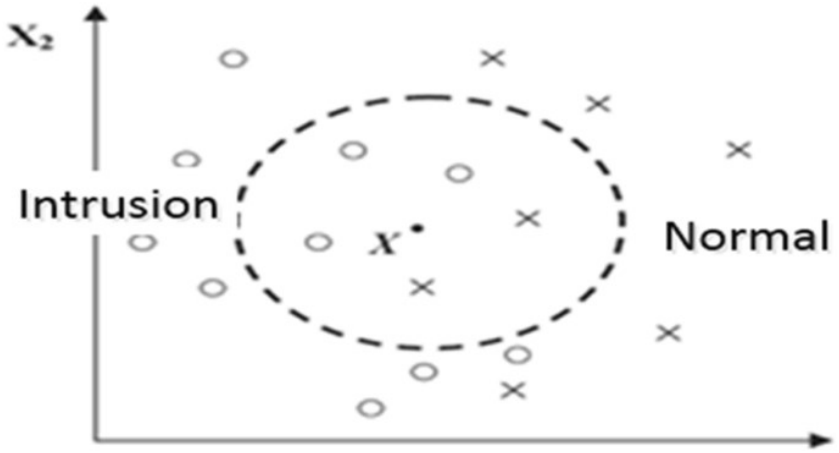


Figure 1. 6 : Un exemple de classification par k-plus proches voisins pour k=5 [6].

### e) Réseau de neurones artificiels

Le concept de réseau de neurones artificiels (ANN) est pris de la matière Biologie où un réseau neuronal joue un rôle important dans notre corps humain. ANN est un système informatique contenant une grande collection d'unités qui sont interconnectées d'une manière qui permet la communication entre les unités. Ces unités sont appelées des nœuds ou des neurones qui sont de simples processeurs fonctionnant en parallèle. Dans ANN, les couches d'entrée sont connectées aux couches cachées qui sont associées à des poids permettant le traitement. Ces couches cachées sont connectées aux couches de sortie pour obtenir les résultats.

### f) Logique floue (Fuzzy logic)

Nous avons deux valeurs de vérité logique, soit 0 ou 1. Mais dans la logique floue, cette plage varie entre 0 et 1 ainsi qu'entre 0 et 1. La logique floue est utilisée à des fins de raisonnement. La logique floue est utilisée dans les IDS d'anomalies.

## B. Classificateurs hybrides

La combinaison d'un ou plusieurs algorithmes d'apprentissage automatique pour améliorer les performances d'un système est appelée classificateur hybride. Les algorithmes supervisés ou non supervisés entrent dans la catégorie des classificateurs hybrides. Ici, un classificateur est conçu sur la base de techniques de regroupement pour éliminer les échantillons étiquetés inconnus afin de reconnaître les motifs. Les algorithmes de classificateurs hybrides sont l'arbre de décision, l'algorithme génétique ou le plus proche voisin et la machine à vecteurs de support. Les classificateurs hybrides combinent les IDS basés sur l'hôte (HIDS) et les IDS basés sur le réseau (NIDS) pour réduire la détection d'utilisation abusive.

### C. Classificateurs ensemblistes

La combinaison de plusieurs "apprenants faibles" pour améliorer les performances du classificateur est appelée classificateur ensembliste. Boosting, stacking et Bagging sont des méthodes pour créer des classificateurs ensemblistes. Certaines applications de classificateurs ensemblistes sont la reconnaissance d'écriture manuscrite, l'évaluation de la qualité des données, la gestion des données de capteurs manquantes, la cartographie des habitats et la prédiction de la croissance des algues. [8]

## 1.4.4 Deep learning (Apprentissage profond)

Le Deep Learning (DL) est un type d'algorithme d'apprentissage automatique avancé utilisé pour résoudre des problèmes complexes et massifs. Basé sur les réseaux de neurones artificiels, le DL est utilisé pour la reconnaissance d'images, la reconnaissance de la parole et le traitement du langage naturel. Les modèles de DL nécessitent souvent des millions de données d'entraînement et prennent du temps pour être entraînés, mais sont utilisés par des entreprises comme Google, Microsoft et Amazon pour alimenter des systèmes entiers tels que les voitures autonomes et les assistants intelligents.

Les deux types de réseaux de neurones profonds utilisés dans le domaine de l'apprentissage en profondeur (deep learning) les plus connus sont :

- **Le réseau de neurones convolutif (CNN)**

Un réseau de neurones convolutif est constitué d'une ou plusieurs couches de convolution, suivies d'une ou plusieurs couches complètement connectées comme dans un réseau de neurones multicouche standard (Vasan et al., 2020b). Un réseau de neurones convolutif contient une couche d'entrée et une couche de sortie, ainsi que plusieurs couches cachées. Les couches cachées d'un CNN contiennent généralement une séquence de couches de convolution qui convoluent avec une multiplication. Un CNN reçoit une entrée 2D et extrait des caractéristiques de haut niveau via une séquence de couches cachées. Les CNN, qui améliorent l'architecture des réseaux de neurones courants, bénéficient des caractéristiques spatiales (Vasan et al., 2020c). Les caractéristiques spatiales sont généralement des types de caractéristiques de trafic appliquées dans le domaine des IDS. Lors de l'application de caractéristiques spatiales, le trafic réseau est transformé en images de trafic ; ensuite, la technique de classification d'image est utilisée pour catégoriser les images de trafic, ce qui permet finalement de détecter le trafic intrusif. Cette technique est relativement récente, mais de nombreuses études récentes prouvent son grand potentiel. Par exemple, Vasan, et al. (Vasan et al., 2020b) ont adopté des techniques de CNN et ont transformé le binaire de logiciel malveillant brut en images en niveaux de gris et en couleur, puis ont appliqué une architecture de CNN fine-tunée.

- **Le réseau de neurones récurrent (RNN)**

Le réseau de neurones récurrent peut fonctionner efficacement sur une série de données de longueurs variables. Cela signifie que les RNN utilisent l'information de leur état antérieur comme entrée pour leur prédiction actuelle, et nous pouvons répéter ce processus pour un nombre arbitraire d'étapes permettant au réseau de propager l'information via son état caché à travers le temps. C'est essentiellement comme donner à un réseau de neurones une mémoire à court terme. Cette fonctionnalité rend les RNN très efficaces pour travailler avec des séquences de données qui se produisent au fil du temps. Yin, et al. (Yin et al., 2017) [9] ont proposé une approche d'apprentissage profond pour la détection d'intrusion en utilisant des réseaux de neurones récurrents (RNN-IDS). Leurs résultats expérimentaux montrent que RNN-IDS est très approprié pour créer des IDS avec une grande précision et que ses performances sont supérieures à celles des méthodes de classification d'apprentissage machine traditionnelles, à la fois pour la classification binaire et multiclasse.

### 1.4.5 Apprentissage automatique vs Apprentissage profond

Les experts en apprentissage automatique et en apprentissage profond n'ont pas encore trouvé de consensus sur ces concepts. Dans ce contexte, de nouvelles idées sont discutées presque chaque jour. L'apprentissage automatique est un concept plus ancien que l'apprentissage profond. L'apprentissage profond peut également être appelé une technique qui effectue de l'apprentissage automatique. Les différences sont énumérées ci-dessous :

1) En apprentissage profond, beaucoup de données sont nécessaires pour amener la structure de l'algorithme à l'idéal. En apprentissage automatique, le problème peut être résolu avec beaucoup moins de données, car la personne donne des fonctionnalités spécifiques à l'algorithme.
2) Les algorithmes d'apprentissage profond tentent d'extraire des fonctionnalités des données. En apprentissage automatique, les fonctionnalités sont déterminées par l'expert.
3) Alors que les algorithmes d'apprentissage profond travaillent sur des machines à haute performance, les algorithmes d'apprentissage automatique peuvent fonctionner sur des CPUs ordinaires.
4) En apprentissage automatique, le problème est généralement divisé en morceaux, ces parties sont résolues les unes après les autres, puis les solutions sont formées en résultat des solutions. En apprentissage profond, le problème est résolu de bout en bout.
5) Il faut beaucoup de temps pour entraîner des algorithmes d'apprentissage profond

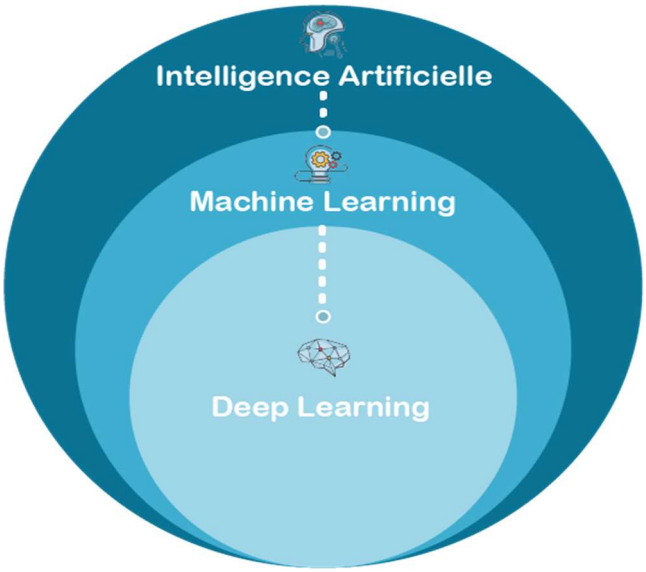


Figure 1. 7 : Différence entre l'apprentissage automatique et l'apprentissage profond [9]

### 1.4.6 Deep Learning pour la détection d'intrusion

Le Deep Learning est une technique d'apprentissage automatique qui peut être utilisée pour la détection d'intrusion sur les réseaux informatiques. Les systèmes de détection d'intrusion basés sur le Deep Learning collectent des données de trafic réseau et utilisent des réseaux de neurones artificiels profonds pour apprendre à détecter les comportements anormaux sur un réseau. Bien que ces systèmes soient plus précis et efficaces que les méthodes traditionnelles, ils nécessitent des quantités massives de données et des ressources de calcul importantes pour le déploiement [9].

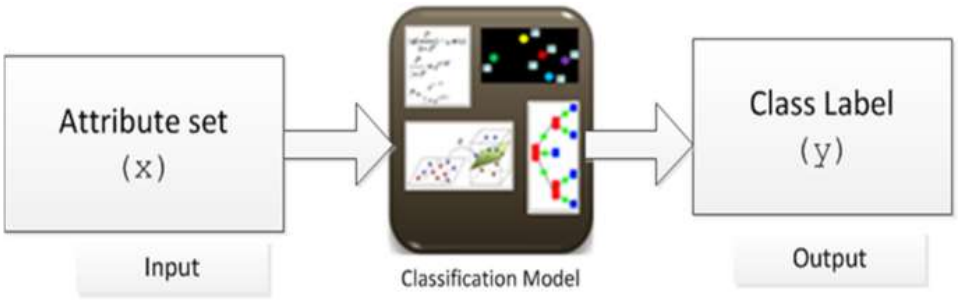


Figure 1. 8: La classification comme tâche [6].

## 1.5 Revue de la littérature

La détection d'intrusions réseau est une problématique majeure en matière de sécurité informatique. Les méthodes traditionnelles basées sur des règles ou des signatures se sont révélées insuffisantes pour lutter contre les attaques de plus en plus sophistiquées. C'est pourquoi l'utilisation de techniques de Machine Learning (ML) et de Deep Learning (DL) a émergé comme une approche prometteuse pour améliorer la détection des intrusions. Plusieurs travaux de recherche ont été menés pour améliorer les méthodes de détection d'intrusions à l'aide de techniques de ML et de DL. Les algorithmes de ML peuvent améliorer considérablement les performances de détection d'intrusions, comme cela a été montré dans plusieurs articles, tels que "A Survey of machine learning for big data processing in intrusion detection systems" (Nguyen et al., 2018) [10] et "Intrusion detection using machine learning techniques : A comprehensive review" (Mirjalili et al., 2021) [11]. En particulier, l'article "Anomaly-based intrusion detection using machine learning techniques : A review" (Idowu et al., 2020) [12] a comparé différentes méthodes de ML pour la détection d'intrusions basées sur les anomalies et a souligné les avantages et les limites de chaque méthode. De même, l'article "Deep learning based intrusion detection systems : A review" (Tavallaee et al., 2019) [13] a présenté les avantages et les inconvénients des différentes approches de DL pour la détection d'intrusions. Il ressort de ces études que les algorithmes d'apprentissage supervisé sont les plus couramment utilisés pour la détection d'intrusions, avec des approches telles que les arbres de décision, les réseaux de neurones et les SVM. Il est également important de souligner l'importance de la qualité des données pour entraîner les modèles de Machine Learning et d'utiliser des approches adaptées aux spécificités de chaque système pour obtenir les meilleurs résultats de détection d'intrusions

## 1.6 Conclusion

En conclusion, ce chapitre nous a permis de passer en revue les différentes approches de détection d'intrusions réseau, en se concentrant sur les techniques classiques ainsi que les principes du Machine Learning appliqués à ce domaine. Nous avons également effectué une revue de la littérature afin de déterminer les travaux de recherche les plus pertinents dans ce domaine. Il est clair que les méthodes de détection d'intrusion basées sur le Machine Learning sont très prometteuses, car elles permettent une détection plus rapide et plus précise des attaques réseau. Cependant, malgré leur efficacité, il reste encore des défis à relever, notamment en termes de la quantité et de la qualité des données d'apprentissage, ainsi que la sélection des paramètres optimaux des algorithmes de Machine Learning.

# Chapitre II
# Méthodologie

# Chapitre II : méthodologie

## 2.1 Introduction

Cette introduction présente la méthodologie utilisée axée sur la détection d'intrusions réseau grâce à l'utilisation du Machine Learning (DL et ML). Le chapitre est structuré en plusieurs étapes clés : présentation des outils et des données, prétraitement des données, sélection des caractéristiques et utilisation de modèles de Machine Learning pour la détection d'intrusion. La base de données NSL KDD est utilisée comme référence pour cette étude. L'objectif est d'améliorer la précision et l'efficacité de la détection d'intrusions en combinant les approches DL et ML, contribuant ainsi à renforcer la sécurité des réseaux informatiques.

## 2.2 Présentation des outils et des données utilisées

### 2.2.1 Présentation des outils utilisés

#### A. Python

Python est un langage de programmation open source, multi-plateformes et orienté objet. Il est largement utilisé dans divers domaines tels que le développement logiciel, l'analyse de données et la gestion d'infrastructures, grâce à sa flexibilité et aux nombreuses bibliothèques spécialisées disponibles. Nous avons utilisé le langage de programmation Python pour implémenter nos modèles de détection d'intrusion. Nous avons bénéficié des bibliothèques présentées sur le tableau ci-dessous, pour développer et évaluer nos modèles :

| Nom de la Bibliothèque | Signification |
|---|---|
| Numpy | Bibliothèque pour effectuer des calculs numériques efficaces sur des tableaux multidimensionnels. |
| Pandas | Bibliothèque pour la manipulation et l'analyse de données. |
| Warnings | Bibliothèque pour gérer les avertissements et les messages d'avertissement émis par le programme. |
| matplotlib.pyplot | Bibliothèque de traçage utilisée pour créer des graphiques et des visualisations. |
| Seaborn | Bibliothèque de visualisation de données basée sur matplotlib, offrant des fonctionnalités supplémentaires et une syntaxe simplifiée. |
| Tensorflow | Bibliothèque d'apprentissage automatique et de deep learning. |
| tensorflow.keras | API d'apprentissage en profondeur intégrée à TensorFlow pour la création de modèles d'apprentissage en profondeur. |
| Xgboost | Bibliothèque d'apprentissage automatique offrant une implémentation optimisée de l'algorithme de boosting utilisant des arbres de décision. |
| scikit-learn | Bibliothèque d'apprentissage automatique offrant une large gamme d'outils pour la préparation des données, la création de modèles, etc. |
| PCA (sklearn.decomposition) | Algorithme d'analyse en composantes principales utilisé pour réduire la dimensionnalité des données. |
| tree (sklearn) | Implémentation des arbres de décision pour la classification et la régression. |
| GaussianNB (sklearn) | Implémentation de l'algorithme de Bayes naïf gaussien pour la classification. |
| LogisticRegression (sklearn) | Implémentation de la régression logistique pour la classification. |
| KNeighborsClassifier (sklearn) | Implémentation de l'algorithme des k plus proches voisins pour la classification. |
| DecisionTreeClassifier (sklearn) | Implémentation des arbres de décision pour la classification. |
| RobustScaler (sklearn.preprocessing) | Transformateur pour mettre à l'échelle les fonctionnalités en utilisant les estimations de la médiane et de l'écart interquartile. |

| | |
|---|---|
| RandomForestClassifier (sklearn.ensemble) | Implémentation des forêts aléatoires pour la classification. |
| RandomForestRegressor (sklearn.ensemble) | Implémentation des forêts aléatoires pour la régression. |
| train_test_split (sklearn.model_selection) | Fonction pour diviser les données en ensembles d'entraînement et de tests. |
| svm (sklearn) | Implémentation des machines à vecteurs de support pour la classification et la régression. |
| metrics (sklearn) | Métriques pour évaluer les performances des modèles d'apprentissage automatique. |
| pd.set_option | Méthode pour définir des options de configuration pour pandas. |
| %matplotlib inline | Directive spécifique à Jupyter Notebook pour afficher les graphiques directement dans le notebook. |
| mpl_toolkits.mplot3d | Une collection de classes d'aide qui facilitent l'affichage de plusieurs images dans Matplotlib. |

Tableau 2. 1 : Bibliothèques utilisées par Python [14].

### B. Jupyter Notebook

Nous avons utilisé Jupyter Notebook comme environnement de développement pour faciliter l'analyse exploratoire des données, l'implémentation des modèles et la visualisation des résultats. Jupyter Notebook une application web open-source qui permet de créer et de partager des documents interactifs appelés "notebooks". Ces notebooks permettent d'exécuter du code en direct, d'afficher des visualisations, d'écrire du texte explicatif et de partager des résultats. Ils sont largement utilisés dans le domaine de la science des données, de l'apprentissage automatique et de la recherche, offrant un environnement convivial pour l'exploration de données, le prototypage de modèles et la communication des résultats. Jupyter Notebook prend en charge plusieurs langages de programmation, notamment Python, R et Julia, et facilite l'intégration de code, de textes descriptifs et d'éléments visuels dans un même document.

### C. Kaggle

"Les données NSL-KDD que nous avons utilisées dans notre étude ont été obtenues à partir de Kaggle, une plateforme en ligne bien connue pour le partage de jeux de données. Kaggle propose un large éventail de jeux de données publiques provenant de divers domaines, y compris la détection d'intrusions réseau.

Une plateforme en ligne dédiée à l'apprentissage automatique (machine learning) et à l'analyse de données. Elle offre aux chercheurs, aux scientifiques des données et aux passionnés un environnement de travail collaboratif pour développer, exécuter et partager du code. Les utilisateurs de Kaggle peuvent utiliser des notebooks Jupyter, qui sont des documents interactifs permettant d'écrire et d'exécuter du code, d'afficher des visualisations et d'ajouter des explications textuelles. Jupyter Notebook est l'un des outils les plus couramment utilisés sur Kaggle pour explorer et analyser des ensembles de données, développer des modèles d'apprentissage automatique et partager des résultats avec la communauté. [14]

### 2.2.2 Présentation des données utilisées

#### 1) Présentation des données utilisées - NSL-KDD

NSL-KDD est un ensemble de données proposé pour résoudre certains des problèmes inhérents à l'ensemble de données KDD'99 mentionnés dans. Bien que cette nouvelle version de l'ensemble de données KDD souffre toujours de certains des problèmes discutés par McHugh et peut ne pas être une représentation parfaite des réseaux réels existants, en raison du manque d'ensembles de données publics pour les systèmes de détection d'intrusions basés sur le réseau, nous pensons qu'il peut encore être utilisé comme un ensemble de données de référence efficace pour aider les chercheurs à comparer différentes méthodes de détection d'intrusions.

#### 2) Avantages de NSL-KDD par rapport à l'ensemble de données KDD original

De plus, le nombre d'enregistrements dans les ensembles d'apprentissages et de tests de NSL-KDD est raisonnable. Cet avantage permet d'exécuter les expériences sur l'ensemble complet sans avoir besoin de sélectionner au hasard une petite partie. Par conséquent, les résultats d'évaluation des différents travaux de recherche seront cohérents et comparables.

#### 3) Utilisation de NSL-KDD pour notre étude de détection d'intrusions

Nous avons utilisé l'ensemble de données NSL-KDD pour notre étude de détection d'intrusions. NSL-KDD est une version améliorée de l'ensemble de données KDDCup'99, largement utilisé dans le domaine de la détection d'anomalies. Il a été choisi pour résoudre les problèmes statistiques rencontrés dans l'ensemble de données KDDCup'99 et fournir un ensemble de données de référence fiable. NSL-KDD présente plusieurs avantages par rapport à l'ensemble de données KDD original. Il élimine les enregistrements redondants dans l'ensemble d'entraînement, permettant une évaluation impartiale des différentes techniques d'apprentissage automatique. De plus, il offre une plus grande variété de taux de classification en fonction de la difficulté, ce qui favorise une évaluation précise. Les ensembles d'entraînement et de test de NSL-KDD sont suffisamment grands pour des expériences complètes.

**4) Utilisation de divers modèles de machine learning et de deep learning**

Nous avons utilisé divers modèles de machine learning et de deep learning, tels que le KNN, le SVM, l'arbre de décision, la forêt aléatoire, le CNN et le RNN, pour détecter les intrusions dans les données NSL-KDD.

**5) Importance de la recherche continue et de l'amélioration des méthodes de détection d'intrusions**

Il est important de noter que l'efficacité des modèles peut varier selon les caractéristiques spécifiques des intrusions et les conditions du réseau. Par conséquent, il est essentiel de poursuivre la recherche et l'amélioration des méthodes de détection d'intrusions en utilisant des ensembles de données tels que NSL-KDD pour des performances optimales.

**6) Utilisation de NSL-KDD pour résoudre les problèmes inhérents à l'ensemble de données KDDCUP'99**

L'ensemble de données NSL-KDD a été suggéré pour résoudre certains des problèmes inhérents à l'ensemble de données KDDCUP'99. KDDCUP'99 est l'ensemble de données le plus largement utilisé pour la détection d'anomalies. Cependant, Travaillée et al ont effectué une analyse statistique sur cet ensemble de données et ont identifié deux problèmes importants qui ont grandement affecté les performances des systèmes évalués, conduisant à une très mauvaise évaluation des approches de détection d'anomalies. Pour résoudre ces problèmes, ils ont proposé un nouvel ensemble de données, NSL-KDD, qui se compose d'enregistrements sélectionnés de l'ensemble de données KDD complet.

- Voici les avantages de NSL-KDD par rapport à l'ensemble de données KDD original :

Tout d'abord, il n'inclut pas d'enregistrements redondants dans l'ensemble d'entraînement, de sorte que les classificateurs ne seront pas biaisés en faveur des enregistrements les plus fréquents. Deuxièmement, le nombre d'enregistrements sélectionnés dans chaque groupe de niveau de difficulté est inversement proportionnel au pourcentage d'enregistrements dans l'ensemble de données KDD d'origine. En conséquence, les taux de classification des différentes méthodes d'apprentissage automatique varient dans une plage plus large, ce qui permet d'obtenir une évaluation précise des différentes techniques d'apprentissage. Troisièmement, le nombre d'enregistrements dans les ensembles d'entraînement et de tests est raisonnable, ce qui permet de mener les expériences sur l'ensemble complet sans avoir besoin de sélectionner aléatoirement une petite partie. Par conséquent, les résultats d'évaluation des différentes recherches seront cohérents et comparables.

**7) Caractéristiques et classes du jeu de données NSL-KDD**

Les données NSL-KDD comprennent 42 caractéristiques et 5 classes qui sont normales et 4 types d'attaques : DoS, Probe, R2L et U2R. L'attaque par déni de service (DoS) est une attaque dans laquelle l'attaquant effectue certaines attaques par déni de service (DoS) consiste à rendre les ressources de calcul

ou de mémoire trop occupées ou trop remplies pour traiter les demandes légitimes, ou à refuser l'accès aux utilisateurs légitimes à une machine. L'attaque de sondage (Probe) est une tentative de recueillir des informations sur un réseau d'ordinateurs dans le but apparent de contourner ses contrôles de sécurité. L'attaque de l'utilisateur au niveau racine (U2R) est une classe d'exploitation dans laquelle l'attaquant commence avec un accès à un compte utilisateur normal sur le système (obtenu peut-être en sniffant des mots de passe, en effectuant une attaque par dictionnaire ou en utilisant l'ingénierie sociale) et est capable d'exploiter une vulnérabilité pour obtenir un accès root au système. L'attaque à distance sur une machine locale (R2L) se produit lorsqu'un attaquant ayant la capacité d'envoyer des paquets à une machine via un réseau, mais n'ayant pas de compte sur cette machine, exploite une vulnérabilité pour obtenir un accès local en tant qu'utilisateur de cette machine.

### 8) Utilisation du jeu de données NSL-KDD dans notre étude de détection d'intrusions

Nous avons utilisé le jeu de données NSL-KDD, largement utilisé dans la communauté de la détection d'intrusion. Ce jeu de données comprend des enregistrements de connexions réseau qui ont été classés comme normaux ou anormaux (intrusions). Il est composé de diverses caractéristiques telles que les protocoles réseau, les types de services, les durées de connexion, les quantités de données transférées, etc. Le jeu de données NSL-KDD contient également des informations sur les types d'attaques présents dans les données, ce qui nous permet de réaliser une détection plus précise des intrusions.

### 9) Division du jeu de données NSL-KDD en ensembles d'entraînement et de test

Nous avons divisé le jeu de données NSL-KDD en ensembles d'entraînement et de tests pour évaluer les performances de nos modèles. L'ensemble d'entraînement a été utilisé pour entraîner les modèles, tandis que l'ensemble de tests a été utilisé pour évaluer leur performance sur des données inconnues.

### 10) Analyse et prédiction des intrusions avec les modèles

Ces modèles nous ont permis d'analyser et de prédire les intrusions de manière efficace et précise.

### 11) Tableau des attributs utilisés de NSL-KDD

| Sr. No. | Feature | Sr. No. | Feature | Sr. No. | Feature |
|---|---|---|---|---|---|
| 1 | Duration | 15 | Su attempted | 29 | Same srv rate |
| 2 | Protocol type | 16 | Num root | 30 | Diff srv rate |
| 3 | Service | 17 | Num file creations | 31 | Srv diff host rate |
| 4 | Flag | 18 | Num shells | 32 | Dst host count |
| 5 | Source bytes | 19 | Num access files | 33 | Dst host srv count |
| 6 | Destination bytes | 20 | Num outbound cmds | 34 | Dst host same srv rate |
| 7 | Land | 21 | Is host login | 35 | Dst host diff srv rate |
| 8 | Wrong fragment | 22 | Is guest login | 36 | Dst host same src port rate |
| 9 | Urgent | 23 | count | 37 | Dst host srv diff host rate |
| 10 | Hot | 24 | Srv count | 38 | Dst host serror rate |
| 11 | Number failed logins | 25 | Serror rate | 39 | Dst host srvserror rate |
| 12 | Logged in | 26 | Srvserror rate | 40 | Dst host rerror rate |
| 13 | Num compromised | 27 | Rerror rate | 41 | Dst host srvrerror rate |
| 14 | Root shell | 28 | Srvrerror rate | 42 | Class label |

Tableau 2. 2 : Tableau des attributs utilisés de NSL-KDD [15].

**12) Chargement des données**

Le jeu de données NSL-KDD est chargé à partir d'un fichier ("KDDTrain+.txt") à l'aide de la fonction **read_csv ()** de Pandas [15].

## 2.3 Prétraitement des données

Le prétraitement des données est une étape essentielle dans le processus d'analyse de données, en particulier dans le domaine de l'apprentissage automatique (machine learning). Avant d'utiliser les données pour entraîner des modèles d'apprentissage automatique, il est crucial de les préparer de manière adéquate afin d'obtenir des résultats précis et fiables. Le prétraitement des données consiste à appliquer une série d'opérations pour nettoyer, transformer et normaliser les données brutes, afin de les rendre adaptées à l'analyse et à l'apprentissage automatique. Un prétraitement approprié permet d'optimiser la qualité des données et d'améliorer les performances des modèles d'apprentissage automatique.

Les étapes de prétraitement souvent utilisées sont :

### 2.3.1 Le nettoyage des données

Le nettoyage des données est le processus de préparation des données pour l'analyse. Celui-ci consiste à supprimer ou corriger les données incorrectes, incomplètes, non pertinentes ou dupliquées dans un ensemble de données. C'est l'une des étapes essentielles de l'apprentissage automatique puisqu'elle joue un rôle clé dans la construction d'un modèle.

### 2.3.2 L'élimination des valeurs aberrantes

L'élimination des valeurs aberrantes en machine learning consiste à détecter et supprimer les valeurs qui diffèrent considérablement des autres dans un ensemble de données. Cela permet d'éviter que ces valeurs atypiques n'affectent négativement les performances du modèle en introduisant des biais ou en faussant les relations entre les variables. L'élimination des valeurs aberrantes améliore la qualité des données utilisées pour l'apprentissage automatique, conduisant à des modèles plus fiables et généralisables.

### 2.3.3 La normalisation des données

La normalisation des données en machine learning est une étape essentielle qui vise à mettre toutes les variables sur une échelle commune. Cela élimine les biais causés par les différences d'échelle et permet aux variables de contribuer équitablement à l'apprentissage du modèle. La normalisation facilite la convergence des algorithmes d'apprentissage, améliore les performances et la robustesse des modèles. Les techniques courantes de normalisation incluent la mise à l'échelle des caractéristiques et la normalisation par z-score. En résumé, la normalisation des données est cruciale pour obtenir des résultats précis et fiables en machine learning.

### 2.3.4 Gestion des valeurs manquantes

La gestion des valeurs manquantes en machine learning est une étape cruciale du prétraitement des données. L'une des approches les plus simples consiste à supprimer les observations (lignes) contenant des valeurs manquantes. Cependant, cette méthode peut entraîner une perte d'informations précieuses. Il est donc important d'évaluer le contexte et la proportion de valeurs manquantes avant de prendre une décision. Dans certains cas, il est préférable d'utiliser des techniques d'imputation, telles que la moyenne, la médiane ou la régression, pour estimer les valeurs manquantes à partir des autres données disponibles. L'objectif est de maintenir l'intégrité des données tout en maximisant les informations utilisées dans le processus d'apprentissage automatique.

### 2.3.5 La réduction de la dimensionnalité

La réduction de la dimensionnalité est une technique utilisée en analyse de données et en apprentissage automatique pour simplifier les ensembles de données en réduisant le nombre de variables ou de dimensions tout en préservant les informations essentielles. L'analyse en composantes principales (PCA) est une méthode courante de réduction de la dimensionnalité. Elle permet de transformer les variables corrélées en variables non corrélées, appelées composantes principales, tout en conservant la variance maximale des données d'origine. La PCA offre des avantages tels que la facilité de visualisation et d'interprétation des données, ainsi que l'amélioration des performances des modèles. Cependant, il est important de normaliser les données avant d'appliquer la PCA et de considérer d'autres approches selon les besoins spécifiques. [16].

### 2.3.6 Prétraitement des données pour le modèle utilisé

**Exploration des données :** Une exploration initiale des données est effectuée à l'aide de méthodes comme describe () pour obtenir des statistiques descriptives sur les données.

**Transformation des variables catégorielles :** Les variables catégorielles telles que 'protocol_type', 'service', et 'flag' sont converties en variables binaires (encodage one-hot) à l'aide de la fonction get_dummies () de Pandas.

**Mise à l'échelle des variables numériques :** Les variables numériques sont mises à l'échelle à l'aide de l'échelle robuste (RobustScaler) pour réduire l'impact des outliers.

**Obtention des noms de colonnes numériques** : Les noms de colonnes numériques sont extraits à partir du DataFrame df_num.

**Normalisation des colonnes numériques :** La fonction **Scaling** est appelée pour normaliser les colonnes numériques du DataFrame **df_num**. Les colonnes normalisées sont stockées dans le DataFrame **scaled_df.**

**Préparation des étiquettes de classification :** Les étiquettes de classe ('outcome') sont converties en 0 pour les exemples normaux et en 1 pour les exemples d'attaque.

**Réduction de dimension :** Une analyse en composantes principales (PCA) est appliquée sur les données pour réduire le nombre de variables à 20 composantes principales.

**Division des données :** Les données sont divisées en ensembles d'entraînement et de test à l'aide de la fonction train_test_split () de Scikit-learn.

Il est important de comprendre que toutes les étapes de prétraitement ne sont pas toujours nécessaires ou appropriées pour tous les ensembles de données. Par exemple, si l'ensemble de données est déjà propre et ne contient pas de valeurs manquantes ou aberrantes, il peut ne pas être nécessaire d'effectuer des étapes de nettoyage supplémentaires.

## 2.4 Sélection des caractéristiques (feature selection)

### 2.4.1 Techniques de sélection de caractéristiques utilisées :

La sélection de caractéristiques (ou features selection) est un processus permettant de choisir les caractéristiques les plus importantes et informatives pour un modèle d'apprentissage automatique. Cela peut être fait pour plusieurs raisons, notamment pour améliorer les performances du modèle, réduire la complexité, réduire le temps de formation et améliorer l'interprétabilité.

a) Analyse en composantes principales (PCA) :

La PCA est une méthode utilisée pour diminuer la dimensionnalité des données en projetant les caractéristiques sur un espace de dimensions plus réduit. Son objectif est de sélectionner un ensemble restreint de caractéristiques qui capturent la plus grande partie de la variance présente dans les données.

b) Importance des caractéristiques pour un arbre de décision :

Les caractéristiques les plus importantes dans un modèle d'arbre de décision sont déterminées en utilisant l'attribut dt.feature_importances_. Cela permet de calculer les importances relatives de chaque caractéristique, et celles qui obtiennent les valeurs les plus élevées sont considérées comme les plus significatives.

c) Importance des caractéristiques pour une forêt aléatoire :

De manière équivalente, les importances des caractéristiques sont évaluées pour une forêt aléatoire en utilisant l'attribut rf.feature_importances_. Une fois de plus, les caractéristiques qui présentent les importances les plus élevées sont considérées comme les plus significatives.

En conclusion, le choix de la technique de sélection de caractéristiques revêt une importance capitale en fonction du problème spécifique, du type de données et des objectifs du modèle d'apprentissage automatique. Chaque méthode présente ses propres avantages et limitations, ce qui nécessite une évaluation minutieuse pour déterminer celle qui convient le mieux à chaque situation.

### 2.4.2 Amélioration des modèles d'apprentissage automatique par la sélection des caractéristiques : Une analyse des données NSL-KDD

La sélection des caractéristiques (ou features selection en anglais) est le processus de choix des variables les plus informatives et pertinentes dans un ensemble de données afin d'améliorer les performances d'un modèle d'apprentissage automatique. Cela consiste à identifier les caractéristiques les plus pertinentes pour prédire ou expliquer une variable cible. La sélection des caractéristiques est une étape cruciale dans le processus de modélisation, car elle permet de réduire la dimensionnalité des données, d'améliorer la précision du modèle, d'accélérer l'apprentissage et de faciliter l'interprétation des résultats. La sélection des caractéristiques dans un ensemble de données dépend généralement de plusieurs facteurs, tels que la pertinence des caractéristiques par rapport à la variable cible, la corrélation entre les caractéristiques, la redondance des informations, la disponibilité des données, ainsi que l'expertise et la connaissance du domaine. Dans le cas spécifique des données NSL-KDD, les caractéristiques sélectionnées peuvent être expliquées comme suit :

**"duration"** : la durée d'une connexion peut être un indicateur important pour la prédiction des intrusions ou des activités malveillantes. Une durée plus longue peut suggérer une tentative d'intrusion plus complexe ou une communication suspecte.

"**protocol_type"** : Le type de protocole utilisé dans une connexion peut être utile pour identifier les différentes formes d'attaques. Différents protocoles peuvent avoir des vulnérabilités spécifiques et des schémas de trafic distincts.

"**service**" : Le service auquel une connexion est associée peut fournir des informations sur la nature de l'activité en cours. Certains services peuvent être plus susceptibles d'être utilisés lors d'attaques ou d'activités malveillantes.

"**flag**" : Le drapeau (flag) indique l'état de la connexion, comme une ouverture, une fermeture ou d'autres états intermédiaires. Les différents drapeaux peuvent révéler des schémas d'activité suspects ou des tentatives d'exploitation de vulnérabilités.

**"src_bytes', 'dst_bytes"** : Ces caractéristiques représentent les nombres d'octets transférés entre l'hôte source et l'hôte de destination. Des valeurs anormalement élevées ou faibles peuvent indiquer des comportements suspects ou des tentatives d'attaques.

**"logged_in"** : Cette caractéristique indique si un utilisateur est connecté ou non. Les activités malveillantes sont souvent associées à des tentatives de connexion non autorisées.

**"count"** et **"srv_count"** : Ces caractéristiques représentent le nombre de connexions vers le même hôte ou le même service au cours d'une période donnée. Des valeurs élevées peuvent indiquer une activité suspecte, telles qu'une numérisation de ports ou une tentative d'attaque par force brute.

**"serror_rate"**, **"rerror_rate"**, **"srv_serror_rate"** et **"srv_rerror_rate"** : Ces caractéristiques représentent les taux d'erreurs lors des connexions, tels que les erreurs de transmission ou les erreurs de protocole. Des taux d'erreur élevés peuvent indiquer des tentatives d'exploitation ou des anomalies.

**"dst_host_count"** et **"dst_host_srv_count"** : Ces caractéristiques représentent le nombre de connexions vers un hôte de destination spécifique ou le nombre de connexions vers un service spécifique sur l'hôte de destination. Des valeurs élevées peuvent indiquer une activité suspecte ou une attaque ciblée [17] .

Le tableau ci-après, regroupe toutes les caractéristiques **"features"** de l'ensemble de données utilisées : tableau :

| **Caractéristique** | **Description** |
|---|---|
| Duration | Durée de la connexion en secondes |
| protocol_type | Type de protocole de la connexion réseau (tcp, udp, etc.) |
| Service | Type de service réseau utilisé |
| Flag | Drapeau de la connexion (SF, S0, REJ, etc.) |
| src_bytes | Nombre d'octets envoyés depuis l'origine vers la destination |
| dst_bytes | Nombre d'octets reçus de la destination vers l'origine |
| Land | Indicateur de la connexion venant de/vers la même adresse/porte source et destination |
| wrong_fragment | Nombre de fragments "wrong" dans les paquets ICMP |
| Urgent | Nombre de paquets urgents |
| Hot | Nombre de "hot indicators" |
| num_failed_logins | Nombre de tentatives de connexion échouées |
| logged_in | Indicateur si l'utilisateur s'est connecté |
| num_compromised | Nombre d'hôtes compromis |
| root_shell | Indicateur si un shell root a été obtenu |
| su_attempted | Indicateur de tentative d'utilisation de la commande su |

| | |
|---|---|
| num_root | Nombre d'opérations réalisées avec les privilèges root |
| num_file_creations | Nombre de créations de fichiers |
| num_shells | Nombre d'ouvertures de shell |
| num_access_files | Nombre d'accès à des fichiers |
| num_outbound_cmds | Nombre de commandes sortantes (obsolète, toujours 0) |
| is_host_login | Indicateur de la connexion à un compte hôte |
| is_guest_login | Indicateur de la connexion en tant qu'invité |
| count | Nombre total de connexions vers le même hôte dans la dernière période de temps |
| srv_count | Nombre total de connexions vers le même service dans la dernière période de temps |
| serror_rate | Taux de connexions avec erreur parmi les connexions de type "SYN" |
| srv_serror_rate | Taux de connexions avec erreur parmi les connexions de type "SYN" vers le même service |
| rerror_rate | Taux de connexions avec erreur |
| srv_rerror_rate | Taux de connexions avec erreur vers le même service |
| same_srv_rate | Taux de connexions vers le même service |
| diff_srv_rate | Taux de connexions vers des services différents |
| srv_diff_host_rate | Taux de connexions vers des hôtes différents pour le même service |
| dst_host_count | Nombre total de connexions vers le même hôte distant dans la dernière période de temps |
| dst_host_srv_count | Nombre total de connexions vers le même service sur l'hôte distant dans la dernière période de temps |
| dst_host_same_srv_rate | Taux de connexions vers le même service sur l'hôte distant |
| dst_host_diff_srv_rate | Taux de connexions vers des services différents sur l'hôte distant |
| dst_host_same_src_port_rate | Taux de connexions depuis le même port source vers le même port de destination sur l'hôte distant |
| dst_host_srv_diff_host_rate | Taux de connexions vers des hôtes différents pour le même service sur l'hôte distant |
| dst_host_serror_rate | Taux de connexions avec erreur parmi les connexions de type "SYN" vers l'hôte distant |
| dst_host_srv_serror_rate | Taux de connexions avec erreur parmi les connexions de type "SYN" vers le même service sur l'hôte distant |

| | |
|---|---|
| dst_host_rerror_rate | Taux de connexions avec erreur vers l'hôte distant |
| dst_host_srv_rerror_rate | Taux de connexions avec erreur vers le même service sur l'hôte distant |
| outcome | Résultat de la connexion (étiquette de classe) |
| level | Niveau de gravité de la connexion |

Tableau 2. 3 : Les caractéristiques 'features' de l'ensemble de données utilisées avec une description [17].

Nous avons travaillé sur 18 "features" de différents types, qui sont regroupées sur le tableau ci-dessous :

| **Type** | **Features** |
|---|---|
| Nominal | Service |
| Binary | logged_in, is_host_login,is_guest_login |
| Numeric | dst_bytes,<br>num_shells, num_outbound_cmds, srv_count, diff_srv_rate,<br>srv_diff_host_rate, dst_host_count, dst_host_srv_count,<br>dst_host_same_srv_rate, dst_host_diff_srv_rate,<br>dst_host_same_src_port_rate, dst_host_srv_diff_host_rate,<br>dst_host_srv_rerror_rate. |

Tableau 2. 4 : Type de caractéristiques sélectionnées [18]

### 2.4.3 Avantages de l'utilisation de la sélection de caractéristiques en apprentissage automatique

- Elle aide à éviter le fléau de la dimensionnalité.
- Elle contribue à la simplification du modèle, facilitant ainsi son interprétation par les chercheurs.
- Elle réduit le temps d'entraînement.
- Elle diminue le surajustement et améliore donc la capacité de généralisation.

## 2.5 Modèles de machine Learning pour la détection d'intrusion

Dans cette section, nous nous intéressons à l'utilisation de modèles de Machine Learning pour renforcer la détection d'intrusion. Nous avons exploré une variété d'algorithmes de Machine Learning, telle que Random Forest, Decision Tree, Régression Logistique, Réseau de Neurones Convolutifs, Réseau de Neurones Récurrents, Machine à Vecteurs de Support), Naive Bayes, XGBoost, Réseaux de Neurones Artificiels et k-plus proches voisins. Chaque algorithme présente des avantages et des caractéristiques uniques pour la détection d'intrusion.

### 2.5.1 Naive Bayes

Le Naïve Bayes est un algorithme d'apprentissage supervisé utilisé pour résoudre des problèmes de classification. Il est basé sur le théorème de Bayes et est principalement utilisé dans la classification de texte avec des ensembles de données volumineux. Le classifieur de Bayes naïf est simple et efficace, permettant de construire rapidement des modèles d'apprentissage automatique capables de faire des prédictions rapides. Il fonctionne sur la base des probabilités des objets et trouve des applications dans des domaines tels que la filtration des spams, l'analyse des sentiments et la classification d'articles. Naive Bayes est utilisé pour la détection d'intrusion. Il classe les données en utilisant des probabilités pour déterminer si elles sont normales ou anormales. C'est un algorithme efficace pour la détection rapide des intrusions. Il est largement utilisé dans les systèmes de sécurité informatique.

**a) Le théorème de Bayes**

Le théorème de Bayes est également connu sous le nom de règle de Bayes ou loi de Bayes, et il est utilisé pour déterminer la probabilité d'une hypothèse en utilisant des connaissances préalables. Il repose sur la probabilité conditionnelle**.** La formule du théorème de Bayes est donnée par l'équation 2.1:

$$\mathrm{P}(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.1}$$

où :

- $P(A|B)$ est la probabilité a posteriori : probabilité de l'hypothèse A étant donné l'événement observé B.
- $P(B|A)$ est la probabilité de vraisemblance : probabilité de la preuve étant donné que l'hypothèse est vraie.
- $P(A)$ est la probabilité a priori : probabilité de l'hypothèse avant d'observer la preuve.
- $P(B)$ est la probabilité marginale : probabilité de la preuve.

**b) Avantages du classifieur Naïve Bayes**

- Naïve Bayes est l'un des algorithmes d'apprentissage automatique rapides et faciles à utiliser pour prédire une classe de données.
- Il peut être utilisé pour des classifications binaires ainsi que pour des classifications multi-classes.
- Il se comporte bien dans les prédictions multi-classes par rapport aux autres algorithmes.
- C'est le choix le plus populaire pour les problèmes de classification de texte.

**c) Inconvénients du classifieur Naïve Bayes**

- Naïve Bayes suppose que toutes les caractéristiques sont indépendantes ou sans lien entre elles, ce qui signifie qu'il ne peut pas apprendre les relations entre les caractéristiques.

### 2.5.2 Les arbres de décision

L'arbre de décision est l'outil le plus puissant et populaire pour la classification et la prédiction. Un arbre de décision est une structure arborescente similaire à un organigramme, où chaque nœud interne représente un test sur un attribut, chaque branche représente un résultat du test, et chaque nœud feuille (nœud terminal) contient une étiquette de classe.

Les arbres de décision sont utilisés pour détecter les intrusions. Ils analysent les données et prennent des décisions basées sur différentes caractéristiques pour identifier les comportements malveillants dans les systèmes informatiques.

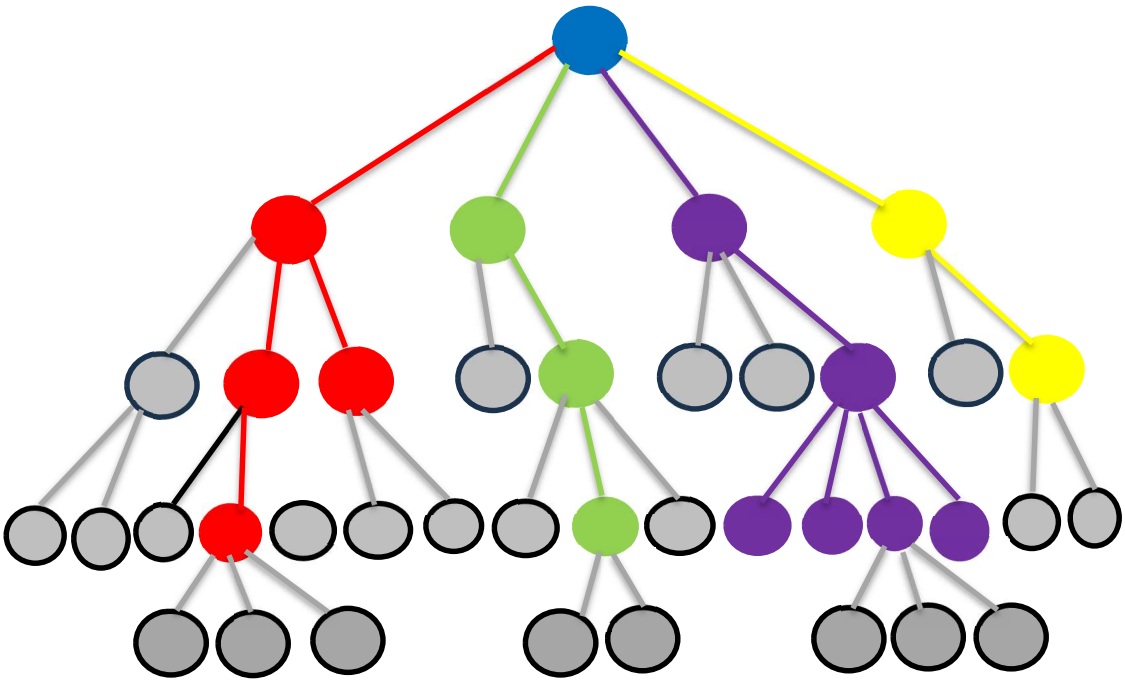

Figure 2. 1 : Les arbres de décision

**a) Avantages de l'arbre de décision**

- Il est simple à comprendre, car il suit le même processus que celui suivi par un être humain lors de la prise de décision dans la vie réelle.
- Il peut être très utile pour résoudre des problèmes liés à la prise de décision.
- Il permet de réfléchir à toutes les issues possibles pour un problème.
- Il nécessite moins de nettoyage des données par rapport à d'autres algorithmes.

**b) Inconvénients de l'arbre de décision**

- L'arbre de décision contient de nombreuses couches, ce qui le rend complexe.
- Il peut présenter un problème de surajustement (overfitting), qui peut être résolu en utilisant l'algorithme de forêt aléatoire (Random Forest).
- Pour davantage de classes, la complexité de calcul de l'arbre de décision peut augmenter.

### 2.5.3 Les machines à vecteurs de support (SVM)

L'algorithme des Machines à Vecteurs de Support (SVM) est largement utilisé en apprentissage automatique pour la classification. Son objectif est de créer un hyperplan, représenté par une ligne ou une frontière de décision, pour séparer les données en différentes classes. Les points extrêmes, appelés vecteurs de support, sont utilisés pour déterminer cet hyperplan. L'algorithme SVM est apprécié pour sa capacité à traiter des problèmes de classification complexes. Les SVM sont utilisés pour la détection d'intrusion. Ils sont capables de séparer efficacement les données en classes, ce qui permet de détecter les comportements malveillants dans les systèmes informatiques. Les SVM sont souvent utilisés en combinaison avec d'autres techniques pour améliorer la précision de la détection d'intrusion.

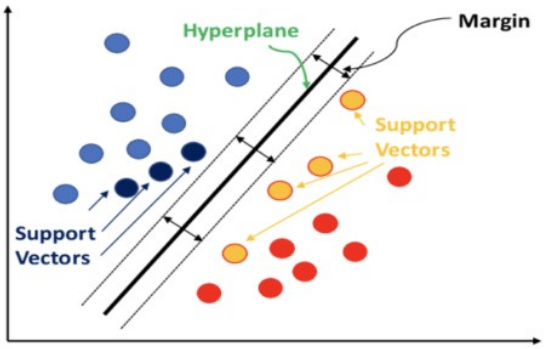


Figure 2. 2 : L'algorithme de Machine à Vecteur de Support (SVM) [19].

**a) Les avantages des machines à vecteurs de support (SVM)**

- Efficaces dans les espaces de dimensions élevées.
- Restent efficaces lorsque le nombre de dimensions est supérieur au nombre d'échantillons.
- Utilisent un sous-ensemble de points d'apprentissage dans la fonction de décision (appelés vecteurs de support), ce qui les rend également efficaces en termes de mémoire.
- Polyvalents : différentes fonctions de noyau peuvent être spécifiées pour la fonction de décision. Des noyaux communs sont fournis, mais il est également possible de spécifier des noyaux personnalisés.

**b) Les inconvénients des machines à vecteurs de support**

- Si le nombre de caractéristiques est beaucoup plus élevé que le nombre d'échantillons, éviter le surajustement en choisissant les fonctions de noyau et le terme de régularisation est crucial.
- Les SVM ne fournissent pas directement des estimations de probabilité, celles-ci sont calculées à l'aide d'une validation croisée à cinq volets coûteux

### 2.5.4 Les Forêts aléatoires (Random Forest)

L'algorithme de Random Forest est une méthode populaire d'apprentissage automatique qui appartient à la technique d'apprentissage supervisé. Il peut être utilisé à la fois pour des problèmes de classification et de régression en apprentissage automatique. Il repose sur le concept d'apprentissage ensembliste, qui consiste à combiner plusieurs classificateurs pour résoudre un problème complexe et améliorer les performances du modèle. Comme son nom l'indique, "Random Forest est un classificateur qui contient plusieurs arbres de décision sur différents sous-ensembles de l'ensemble de données donné et fait la moyenne pour améliorer la précision prédictive de cet ensemble de données". Au lieu de s'appuyer sur un seul arbre de décision, la forêt aléatoire prend la prédiction de chaque arbre et, sur la base du vote majoritaire des prédictions, elle prédit la sortie finale. Un plus grand nombre d'arbres dans la forêt conduit à une plus grande précision et évite le problème du surajustement. Le Random Forest est une méthode utilisée pour détecter les intrusions. Elle consiste à combiner plusieurs arbres de décision afin d'améliorer la précision de la détection. Cette approche utilise un vote majoritaire pour classer les comportements comme étant malveillants ou non.

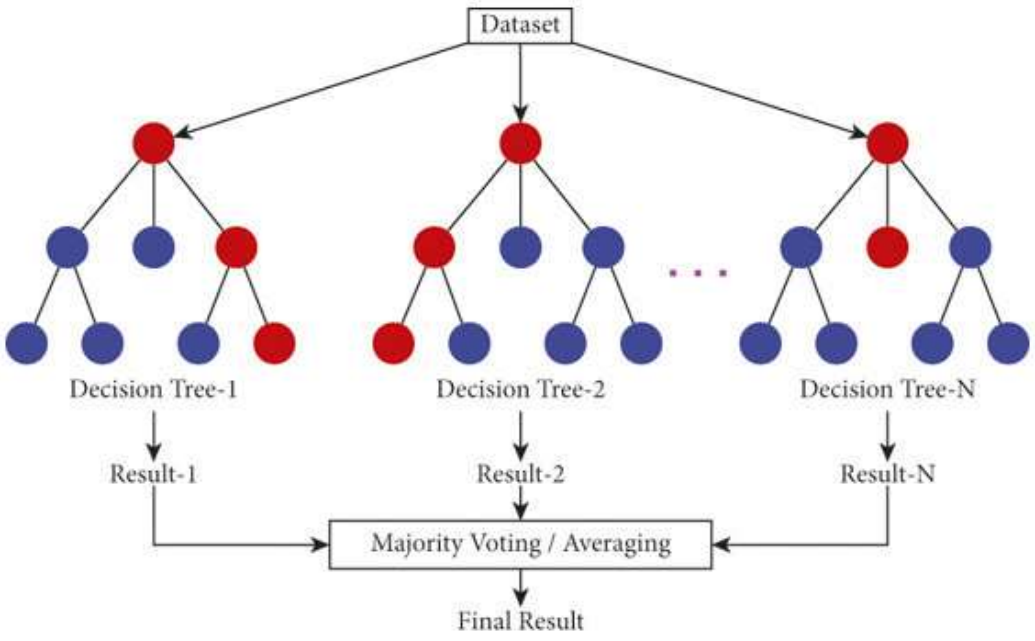


Figure 2. 3 : L'algorithme de Random Forest [20]

**a) Avantages de Random Forest**

- Random Forest est capable d'effectuer à la fois des tâches de classification et de régression.
- Il peut gérer de grands ensembles de données avec une haute dimensionnalité.
- Il améliore la précision du modèle et évite le problème du surajustement.

**b) Inconvénients de Random Forest**

- Bien que Random Forest puisse être utilisé pour des tâches de classification et de régression, il n'est pas aussi adapté aux tâches de régression.

### 2.5.5 Régression logistique

La régression logistique est un algorithme d'apprentissage automatique utilisé pour prédire une variable dépendante catégorique à partir de variables indépendantes. Elle utilise une fonction logistique en forme de "S" pour prédire des valeurs probabilistes entre 0 et 1. Contrairement à la régression linéaire, elle est utilisée pour résoudre des problèmes de classification plutôt que des problèmes de régression. Elle offre la possibilité de fournir des probabilités et de classifier de nouvelles données. La régression logistique est utilisée pour déterminer les variables les plus efficaces dans la classification des observations. La régression logistique est utilisée dans la détection d'intrusion pour analyser les données réseau et prédire si un événement est une intrusion en se basant sur des variables indépendantes. Elle permet d'identifier les comportements anormaux et de détecter les intrusions dans les systèmes informatiques.

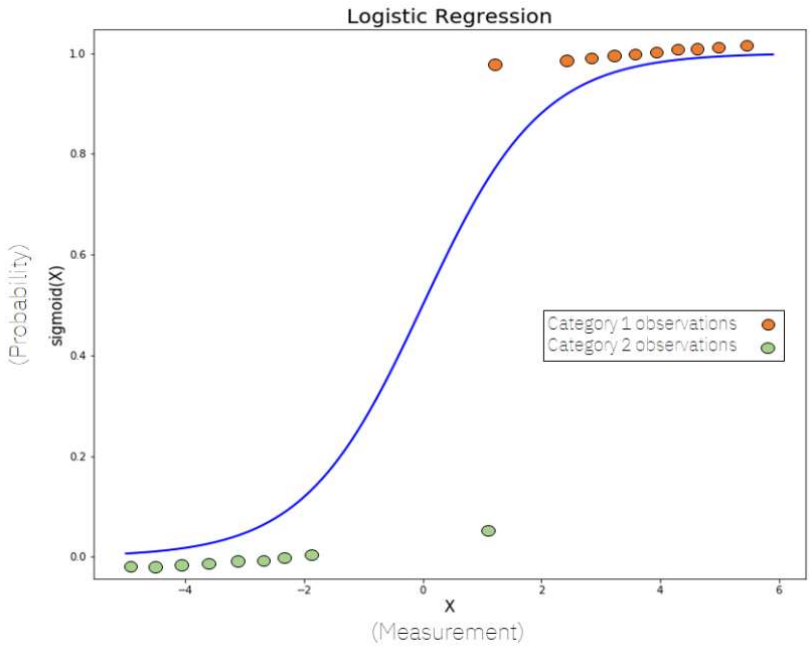


Figure 2. 4 : La régression logistique [21]

**a) Avantages de régression logistique**

- La régression logistique est largement utilisée, car elle est extrêmement efficace et ne requiert pas d'énormes quantités de ressources informatiques. Elle peut être interprétée facilement et ne nécessite pas de mise à l'échelle des caractéristiques d'entrée. Elle est simple à régulariser et les résultats qu'elle fournit sont des probabilités prédites bien calibrées.
- Tout comme dans la régression linéaire, la régression logistique a tendance à fonctionner plus efficacement lorsque les attributs non liés à la variable de sortie et ceux qui sont corrélés sont omis. L'ingénierie des caractéristiques joue un rôle important dans l'efficacité des performances de la régression logistique et linéaire.
- La régression logistique est également simple pour former les utilisateurs et facile à mettre en œuvre, ce qui en fait une excellente référence pour aider à mesurer les performances d'autres algorithmes complexes.

**b) Inconvénients de régression logistique**

- La régression logistique présente quelques limitations. Elle n'est pas adaptée pour résoudre des problèmes non linéaires, ce qui peut être une limitation dans de nombreux systèmes actuels. De plus, d'autres algorithmes plus puissants sont disponibles, capables de fournir des prédictions plus précises et complexes que la régression logistique.
- La régression logistique est sensible à la qualité et à la représentation des données. Si des variables indépendantes importantes sont manquantes, les résultats peuvent être peu fiables. De plus, la régression logistique est limitée à la prédiction de résultats catégoriques, car elle génère des résultats discrets. Enfin, l'algorithme est susceptible de sur ajuster les données, ce qui peut affecter sa capacité à généraliser correctement.

### 2.5.6 K-Nearest Neighbors

L'algorithme des k-plus proches voisins (KNN) est une méthode simple et efficace d'apprentissage automatique supervisé. Il classe un nouvel échantillon en le comparant aux échantillons existants et en l'assignant à la catégorie la plus similaire. KNN est utilisé pour la classification et la régression, sans faire d'hypothèses sur les données. C'est un algorithme paresseux, stockant les données d'entraînement et les utilisant lors de la phase de classification. Il est largement utilisé pour résoudre des problèmes de classification dans de nombreux domaines. La méthode KNN est utilisée pour la détection d'intrusion en analysant les caractéristiques des données réseau. Elle classe les nouveaux échantillons en les comparant aux échantillons existants et les assigne à la catégorie la plus similaire. Cela permet d'identifier les comportements anormaux et de détecter les intrusions dans les systèmes informatiques.

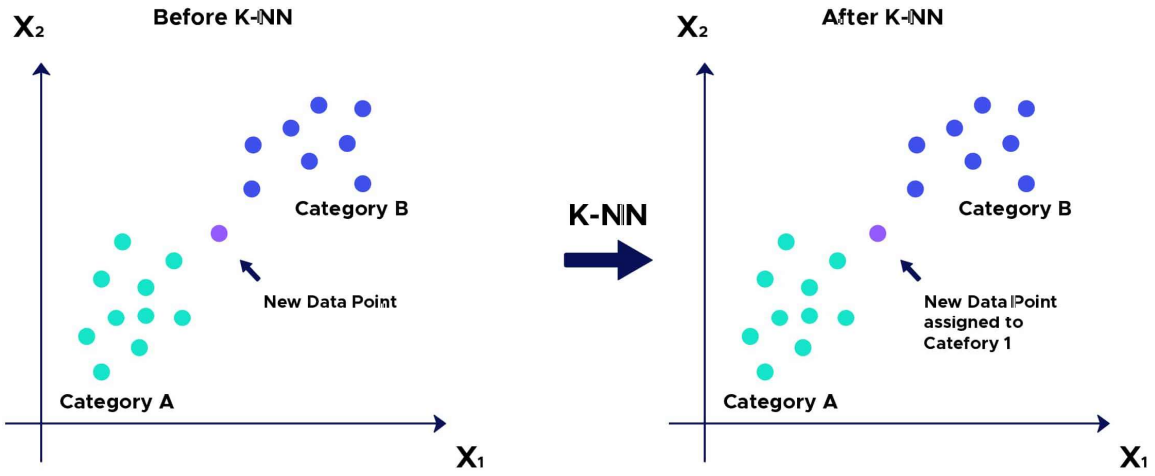


Figure 2. 5 : L'algorithme des k-plus proches voisins (KNN) [22].

**a) Sélection de la valeur K :**

Voici quelques points à prendre en compte lors de la sélection de la valeur de K dans l'algorithme KNN :

- Il n'y a pas de méthode particulière pour déterminer la meilleure valeur de "K", donc nous devons essayer différentes valeurs pour trouver la meilleure. La valeur préférée pour K est généralement 5.
- Une valeur très basse pour K, comme K=1 ou K=2, peut être bruitée et être sensible aux effets des valeurs aberrantes dans le modèle.
- Des valeurs élevées pour K sont bonnes, mais elles peuvent présenter certaines difficultés.

**b) Avantages de l'algorithme KNN**

- Il est simple à implémenter.
- Il est robuste face aux données d'entraînement bruitées.
- Il peut être plus efficace lorsque les données d'entraînement sont nombreuses

**c) Inconvénients de l'algorithme KNN**

- Il est nécessaire de déterminer la valeur de K, ce qui peut parfois être complexe.
- Le coût de calcul est élevé en raison du calcul de la distance entre les points de données pour tous les échantillons d'entraînement.

### 2.5.7 XGBoost (eXtreme Gradient Boosting)

XGBoost (eXtreme Gradient Boosting) est un algorithme d'apprentissage automatique basé sur des arbres de décision. Il utilise la technique de boosting pour entraîner de manière séquentielle plusieurs modèles afin de corriger les erreurs des modèles précédents. XGBoost optimise efficacement le gradient et utilise des techniques d'élagage et de régularisation pour améliorer les performances et éviter le surajustement. Il est largement utilisé dans divers domaines et est apprécié pour sa précision et sa performance élevées. XGBoost est un algorithme utilisé dans la détection d'intrusion en analysant les caractéristiques des données réseau. Il permet de créer un modèle prédictif performant pour détecter les comportements anormaux et les intrusions. En utilisant des arbres de décision et des techniques de boosting, XGBoost améliore la précision de la détection.

**a) Avantages de l'algorithme XGBoost**

- Haute performance : XGBoost est reconnu pour sa vitesse d'exécution élevée et son efficacité grâce à des techniques d'optimisation avancées.
- Précision prédictive : XGBoost se distingue par sa précision élevée dans la classification et la régression, gérant efficacement les ensembles de données complexes pour des modèles prédictifs de qualité.
- Gestion des données manquantes : XGBoost excelle dans la gestion des données manquantes, offrant un avantage majeur par rapport à d'autres algorithmes d'apprentissage automatique.

**b) Inconvénients de l'algorithme XGBoost**

- Complexité : L'algorithme XGBoost est complexe et nécessite une expertise pour une implémentation correcte, ce qui peut être difficile pour les débutants.
- Temps de calcul : Bien que XGBoost soit rapide, il peut prendre plus de temps à s'exécuter que des méthodes plus simples, ce qui peut être un inconvénient pour de très grands ensembles de données.
- Sensibilité aux paramètres : La sélection des paramètres dans XGBoost peut être un défi, nécessitant une expérimentation et une validation approfondies pour obtenir les meilleurs résultats [23].

### 2.5.8 Réseaux de neurones artificiels

Les réseaux neuronaux artificiels (RNA) sont des modèles computationnels inspirés du fonctionnement du cerveau humain. Ils sont composés de neurones artificiels interconnectés qui permettent de traiter des informations et d'effectuer des tâches d'apprentissage automatique. Les RNA sont utilisés pour résoudre une variété de problèmes tels que la classification, la régression, la reconnaissance de formes, et bien plus encore. La détection d'intrusion utilisant les réseaux neuronaux artificiels (RNA) consiste à entraîner un modèle de RNA pour analyser les données réseau et détecter les comportements anormaux, permettant ainsi d'identifier les intrusions de manière efficace et précise.

**a) Les avantages des réseaux neuronaux artificiels (RNA)**

- Capacité à apprendre à partir de données complexes et non linéaires
- Flexibilité pour traiter différents types de problèmes (classification, régression, etc.)
- Capacité à effectuer des prédictions précises une fois entraînées correctement.

**b) Les inconvénients des RNA**

- Sensibilité aux surajustements.
- Besoin de grandes quantités de données d'entraînement .
- Temps de calcul potentiellement long.
- Manque de transparence et d'interprétabilité par rapport à d'autres algorithmes [24].

### 2.5.9 Les réseaux de neurones récurrents (RNN)

Les réseaux de neurones récurrents (RNN) sont des architectures de réseaux neuronaux qui sont capables de traiter des données séquentielles en prenant en compte la dépendance temporelle entre les éléments successifs de la séquence. Ils sont particulièrement efficaces pour des tâches telles que la prédiction de séquences, la traduction automatique et l'analyse de sentiments. La détection d'intrusion utilisant des réseaux de neurones récurrents (RNN) permet d'analyser les séquences de données réseau pour détecter les comportements anormaux et identifier les intrusions de manière efficace.

- **Les réseaux LSTM (Long Short-Term Memory)**

Également connus sous le nom de réseaux de mémoire à court et long terme, est une forme spéciale de réseaux de neurones récurrents capables d'apprendre des dépendances à long terme.

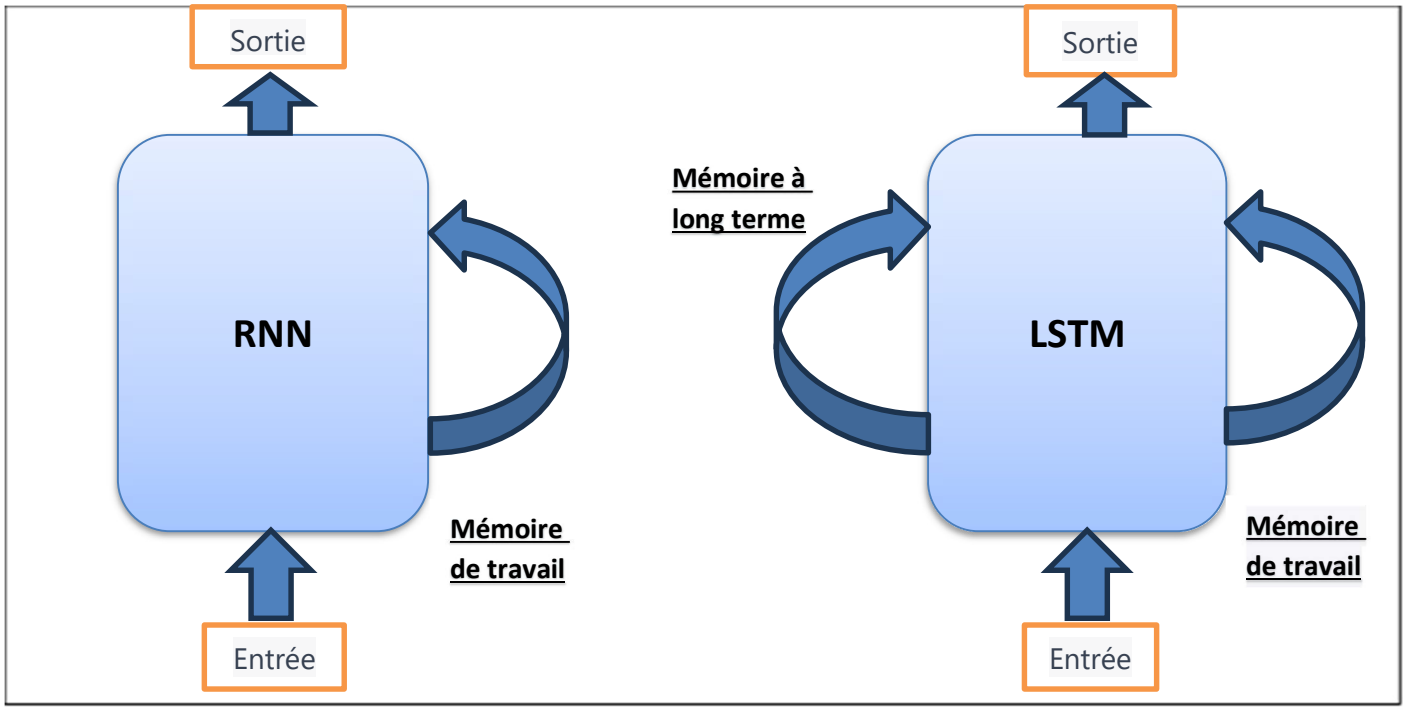


Figure 2. 6 : Analyse comparative des RNN avec LTSM.

**a) Avantages des réseaux RNN**

- Capacité à traiter des séquences de longueur variable et à capturer des dépendances à long terme.
- Adaptabilité aux données temporelles et capacité à modéliser des dynamiques temporelles complexes.

**b) Inconvénients des réseaux RNN**

- Difficulté à apprendre des dépendances à long terme dans des séquences très longues.
- Vulnérabilité au problème du gradient qui peut entraîner des difficultés lors de l'entraînement sur des séquences longues [25].

## 2.5.10 Les réseaux de neurones convolutifs (CNN)

Les réseaux de neurones convolutifs (CNN) sont une classe spécifique de réseaux de neurones artificiels, souvent utilisés pour l'apprentissage automatique supervisé dans le domaine de la vision par ordinateur. Ils sont particulièrement adaptés pour traiter des données structurées telles que des images.

a) Détection d'intrusion réseau avec des CNN

Dans le contexte de la détection d'intrusion réseau, les réseaux de neurones convolutifs peuvent être utilisés pour analyser le trafic réseau et détecter des modèles ou des comportements suspects qui pourraient indiquer une activité malveillante.

b) Fonctionnement des réseaux de neurones convolutifs

Le fonctionnement d'un réseau de neurones convolutif repose sur deux opérations principales : la convolution et le pooling.

La convolution implique l'application d'un filtre (ou noyau) à l'image ou au signal d'entrée pour extraire des caractéristiques spécifiques.

Le pooling, quant à lui, réduit la dimensionnalité des caractéristiques extraites en prenant les valeurs maximales ou moyennes dans des régions d'intérêt.

c) Fonctions d'activation dans les réseaux de neurones convolutifs

Les réseaux de neurones convolutifs utilisent également des fonctions d'activation pour introduire une non-linéarité dans le modèle. Des fonctions couramment utilisées sont la fonction ReLU (Rectified Linear Unit) ou la fonction sigmoïde, qui permettent de modéliser des relations complexes entre les entrées et les sorties du réseau [26].

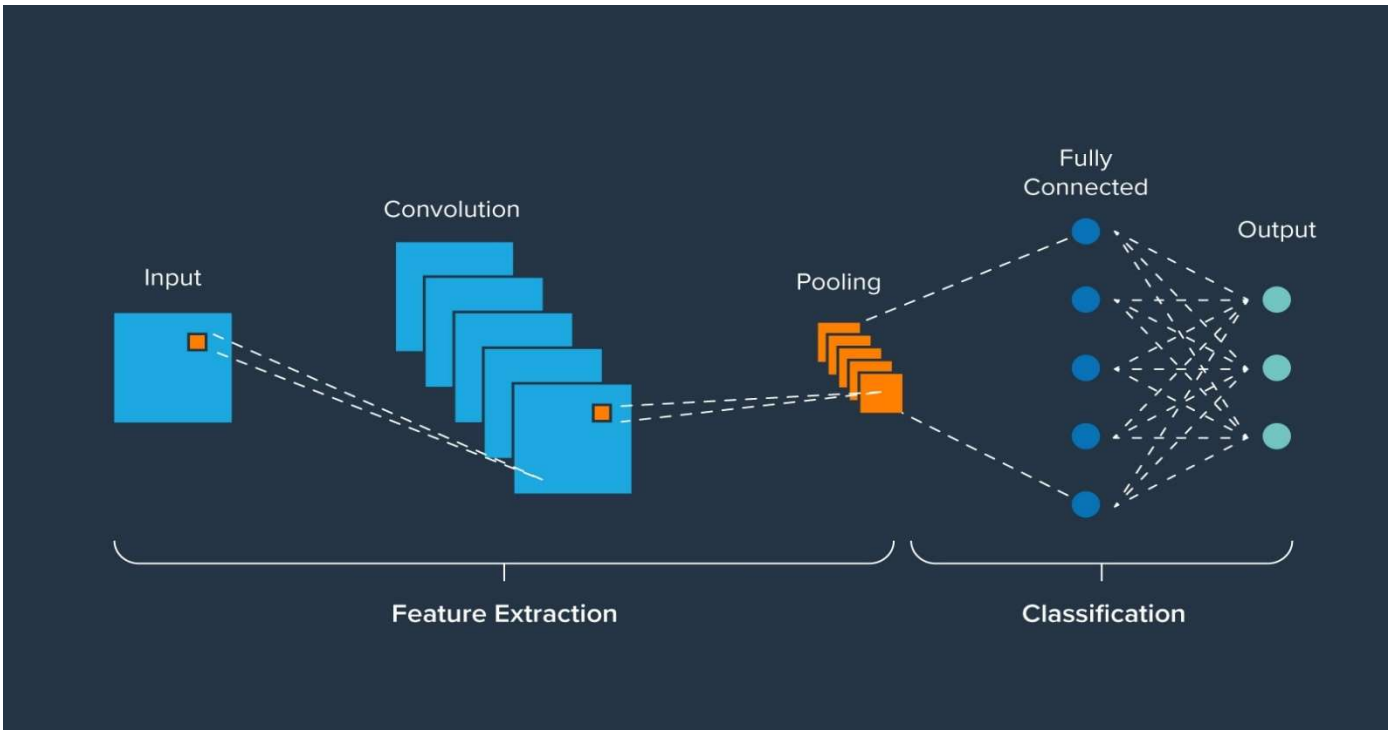


Figure 2. 7 : Exploration de l'Architecture Révolutionnaire des Réseaux de Neurones Convolutifs (CNN) [27].

**d) Avantages des réseaux de neurones convolutifs (CNN)**

- Capacité à capturer des caractéristiques complexes et à extraire des motifs significatifs à partir des données.
- Performances élevées dans les tâches de vision par ordinateur telles que la reconnaissance d'images et la détection d'objets.
- Capacité à gérer efficacement des données de grande dimension et de grande complexité.

**e) Inconvénients des réseaux de neurones convolutifs (CNN)**

- Nécessité d'un grand ensemble de données pour un entraînement efficace.
- Requiert des ressources informatiques importantes en termes de puissance de calcul et de mémoire.
- Sensible à l'overfitting (surapprentissage) lorsque les données d'entraînement sont insuffisantes [28].

## 2.6 Conclusion

En conclusion, ce chapitre présente une méthodologie pour la détection d'intrusions réseau en utilisant le Machine Learning, en combinant les approches de Deep Learning (DL) et de Machine Learning traditionnel (ML). L'utilisation de la base de données NSL KDD comme référence permet d'obtenir des résultats pertinents dans le domaine de la détection d'intrusions réseau. L'importance du prétraitement des données et de la sélection des caractéristiques est soulignée pour améliorer les performances des modèles de Machine Learning. Ce chapitre pose les bases d'une approche solide pour renforcer la sécurité des réseaux informatiques en détectant de manière plus précise les activités suspectes.

# Chapitre III
# Résultats et analyse

# Chapitre III : Résultats et analyses

## 3.1 Introduction

Ce chapitre présente les résultats et les analyses de notre étude sur la détection d'intrusions réseau en utilisant le Machine Learning et le Deep Learning. Les performances des modèles sont évaluées et comparées en fonction de mesures telles que l'exactitude, la précision, le rappel, tout en considérant les caractéristiques des données et des critères tels que la complexité, la capacité en temps réel, l'implémentation et la robustesse aux données bruyantes. Les résultats et les analyses contribuent à l'avancement de la détection d'intrusions réseau et guident les décisions en matière de sécurité informatique.

## 3.2 Présentation des résultats obtenus

### 3.2.1 Les métriques de performances

Il y a 4 termes importants qui sont des termes utilisés dans l'évaluation des performances des modèles de classification :

**Vrais Positifs** : Les cas où nous avons prédit OUI et que la sortie réelle était également OUI.

**Vrais Négatifs** : Les cas où nous avons prédit NON et que la sortie réelle était également NON.

**Faux Positifs** : Les cas où nous avons prédit OUI et que la sortie réelle était NON.

**Faux Négatifs** : Les cas où nous avons prédit NON et que la sortie réelle était OUI.

Ces termes sont souvent utilisés pour calculer différentes mesures d'évaluation de la performance d'un modèle, telles que la précision, le rappel (sensibilité), l'exactitude, dans le domaine de l'apprentissage automatique et de l'évaluation des modèles de classification.

L'exactitude (accuracy) : est le rapport entre le nombre total de prédictions correctes et le nombre total d'échantillons dans l'ensemble de données.

Mathématiquement, accuracy peut être calculée de la manière suivante :

$$\text{Exactitude} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{FP} + \text{TN} + \text{FN})} \tag{3.1}$$

Précision : Il s'agit du nombre de résultats positifs corrects divisé par le nombre de résultats positifs prédits

par le classifieur.

Mathématiquement, la précision peut être calculée de la manière suivante :

$$\text{Précision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{3.2}$$

Rappel : C'est le nombre de résultats positifs corrects divisé par le nombre total d'échantillons pertinents (tous les échantillons qui auraient dû être identifiés comme positifs). [29]

Mathématiquement, le rappel peut être calculé de la manière suivante :

$$\text{Rappel} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{3.3}$$

### 3.2.2 Analyse des métriques de performances

**1)** Le tableau suivant présente les performances des différents algorithmes utilisés dans notre étude sur la détection d'intrusions réseau. Les métriques clés telles que la précision d'entraînement, la précision de test, la précision de rappel et le rappel sont présentées pour chaque algorithme. Les performances ont été évaluées à l'aide des ensembles d'entraînement et de tests, et les résultats obtenus démontrent l'efficacité des différents modèles dans la détection des intrusions. Analysons de plus près les performances de chaque algorithme.

| Algorithmes / Métriques | Training Accuracy | Test Accuracy | Training Precision | Test Precision | Training Recall | Test Recall |
|---|---|---|---|---|---|---|
| **RandomForestClassifier** | 99.994% | 99.881% | 99.994% | 99.941% | 99.994% | 99.805% |
| **SVM (Linear SVC)** | 96.901% | 96.733% | 97.502% | 97.191% | 95.784% | 95.800% |
| **Logistic Regression** | 87.758% | 87.517% | 86.648% | 86.687% | 87.068% | 86.680% |
| **KNN (KNeighborsClassifier)** | 99.052% | 98.936% | 99.225% | 99.056% | 98.731% | 98.671% |
| **DecisionTreeClassifier** | 99.994% | 99.877% | 100.000% | 99.848% | 99.987% | 99.890% |
| **Naive Bayes** | 91.802% | 91.605% | 92.626% | 92.532% | 89.479% | 89.296% |
| **Réseau de neurones artificiels** | 97.826% | 97.729% | 98.990% | 99.358% | 96.360% | 95.774% |
| **XGBoost** | 99.991% | 99.916% | 99.989% | 99.932% | 99.991% | 99.889% |
| **Réseau de neurones convolutionnels** | 99.357% | 99.412% | 99.294% | 99.434% | 99.184% | 99.387% |
| **Réseau de neurones récurrents** | 99.289% | 99.239% | 99.239% | 99.294% | 99.1846% | 99.1845% |

Tableau 3. 1 : Performance des modèles de détection d'intrusions réseau

**2)** Le diagramme compare les performances de différents algorithmes d'apprentissage automatique sur plusieurs caractéristiques. Les algorithmes évalués comprennent RandomForest, SVM, Logistic Regression, KNN, Decision Tree, Naive Bayes, ANN, XGBoost, CNN et RNN. Les barres représentent chaque algorithme avec des couleurs spécifiques à chaque caractéristique. Les hauteurs des barres indiquent les pourcentages de performance obtenus. Ce diagramme facilite la comparaison des performances et l'identification des meilleurs algorithmes. En résumé, il offre un aperçu clair des performances relatives des algorithmes évalués.

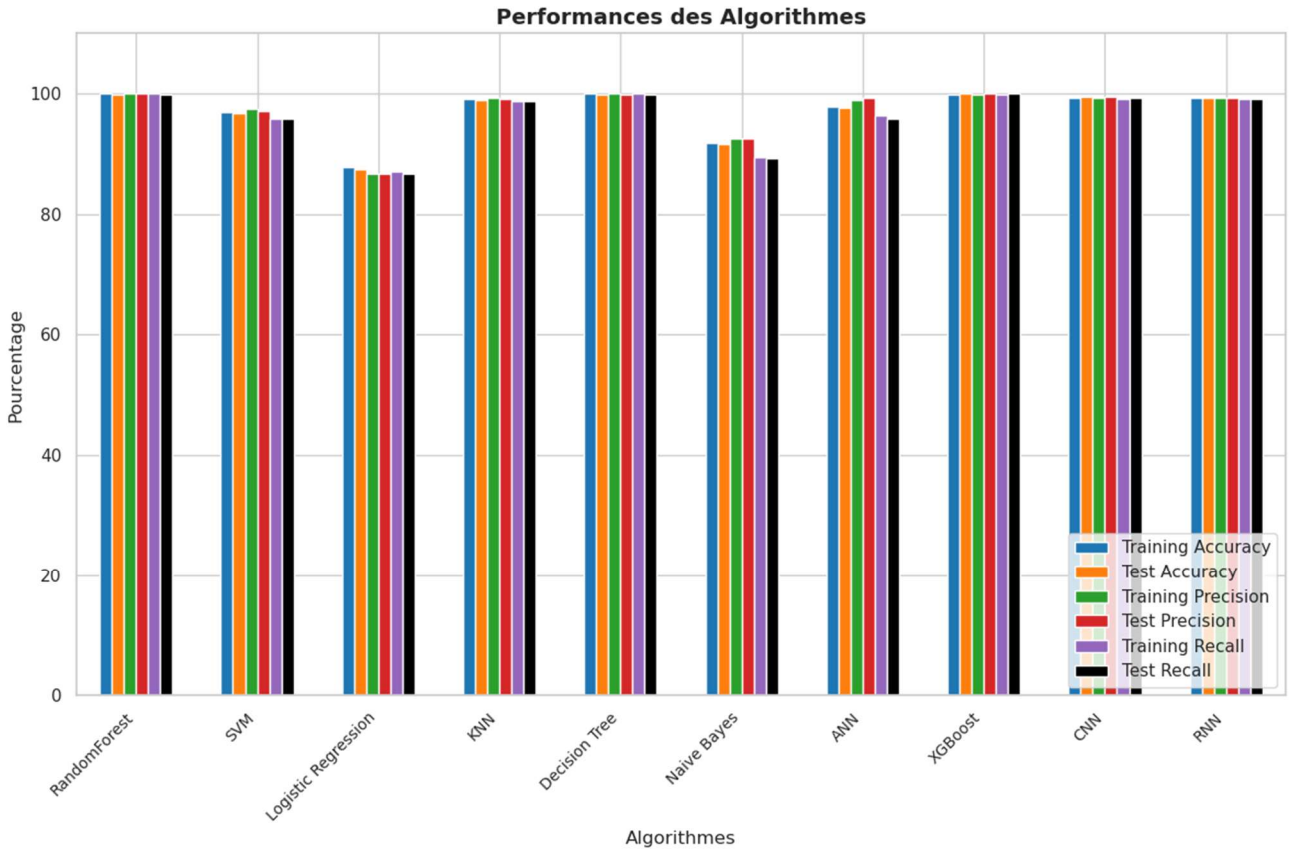


Figure 3. 1 : Analyse comparative des performances des algorithmes de machine learning.

### 3.2.3 Matrice de confusion

#### a) logistique régression

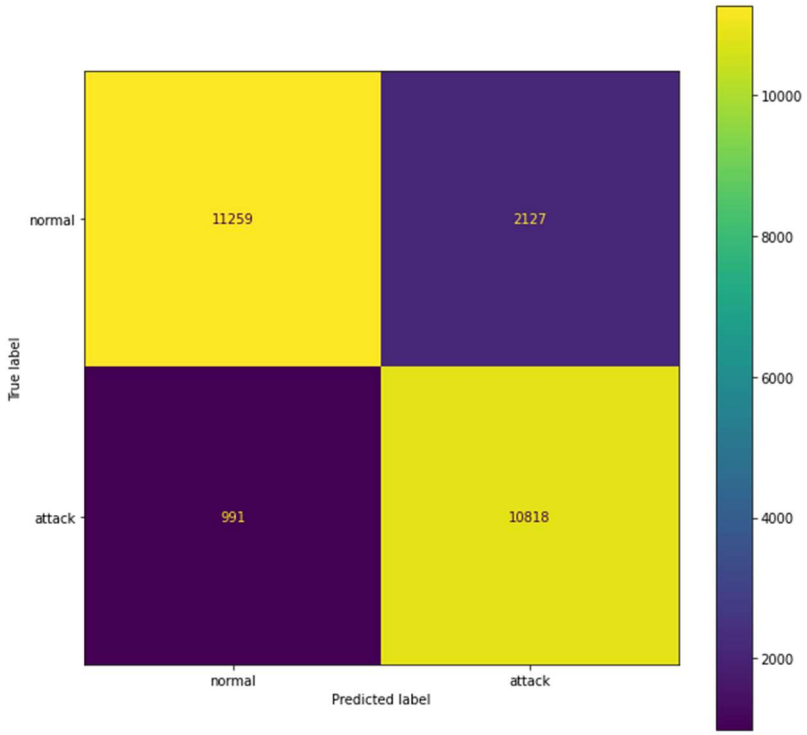


Figure 3. 2 : Matrice de Confusion de la Régression Logistique pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de la Régression Logistique pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 12 535, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle de régression logistique.

False Positives (FP) : Avec une valeur de 851, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle.

False Negatives (FN) : Avec une valeur de 1 264, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle.

True Negatives (TN) : Avec une valeur de 10 545, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle de régression logistique.

### b) KNN (k-nearest neighbors)

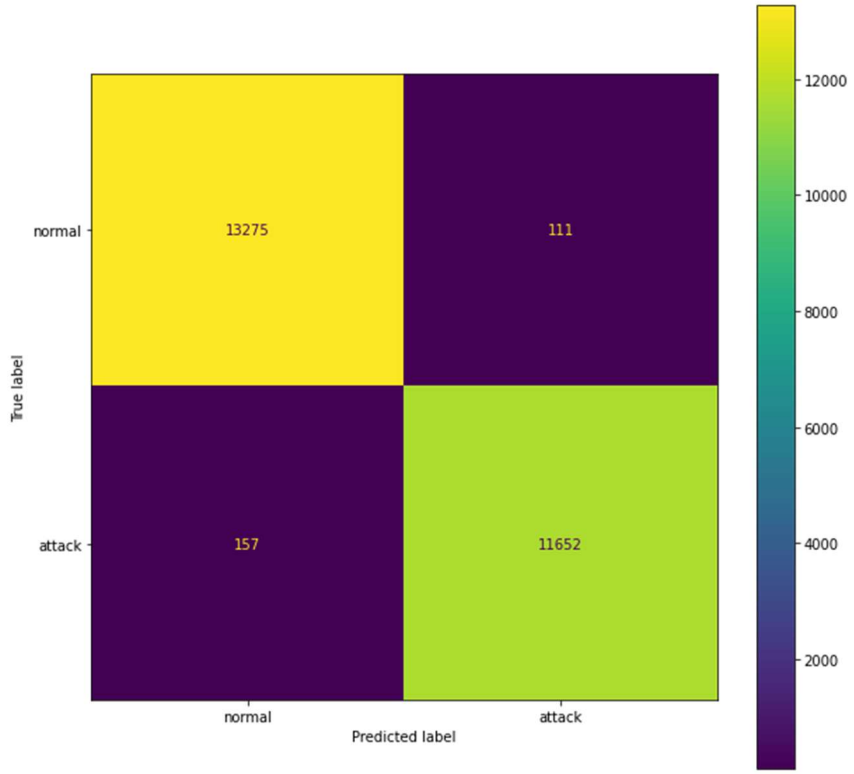


Figure 3. 3: Matrice de Confusion de la KNN (k-nearest neighbors) pour la Classification (Attack, Normal).

**Explication Matrice de Confusion de la KNN (k-nearest neighbors) pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 275, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle KNN.

False Positives (FP) : Avec une valeur de 111, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle.

False Negatives (FN) : Avec une valeur de 157, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle.

True Negatives (TN) : Avec une valeur de 11 652, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle KNN.

### c) Naive Bayes

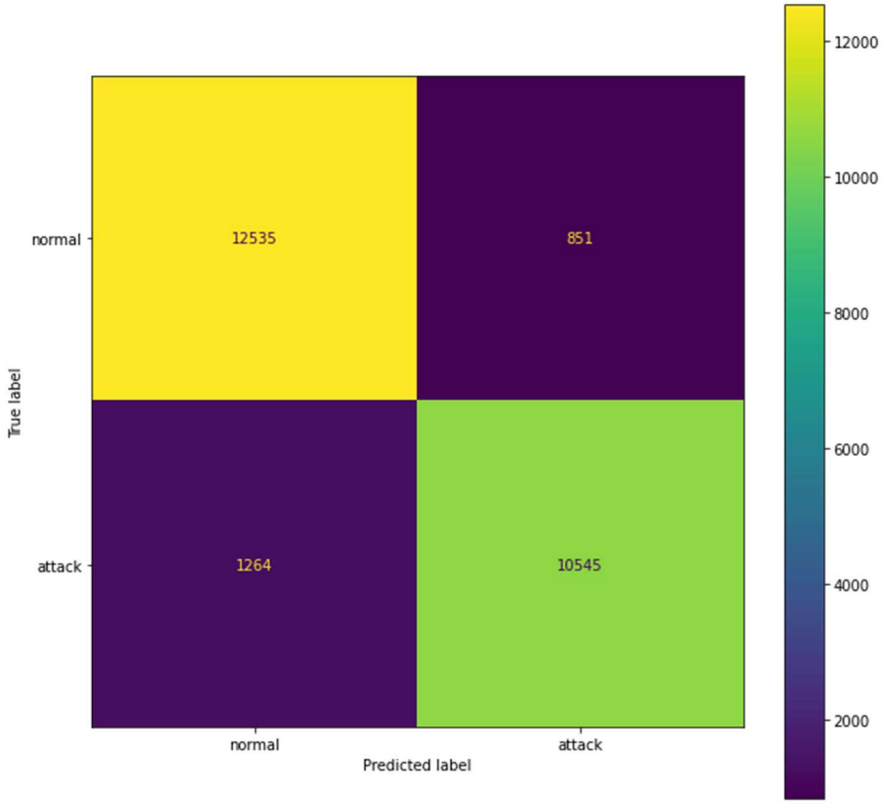


Figure 3. 4: Matrice de Confusion de Naive Bayes pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de la Naive Bayes pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 12 535, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle Naive Bayes.

False Positives (FP) : Avec une valeur de 851, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle.

False Negatives (FN) : Avec une valeur de 1 264, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle.

True Negatives (TN) : Avec une valeur de 10 545, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle Naive Bayes.

### d) SVM (Support Vector Machine)

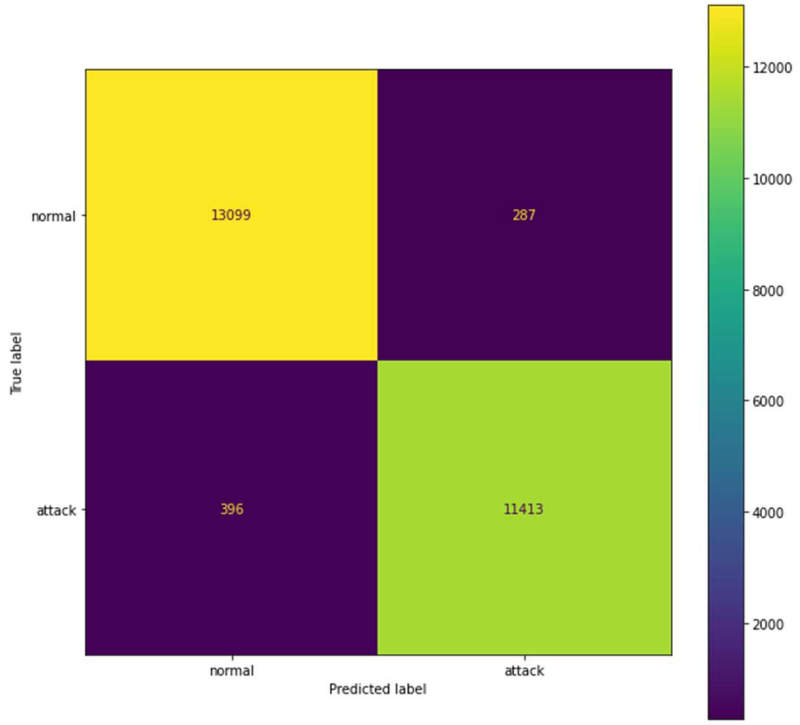


Figure 3. 5 : Matrice de Confusion de SVM (Support Vector Machine) pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de la SVM pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 099, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle SVM.

False Positives (FP) : Avec une valeur de 287, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle.

False Negatives (FN) : Avec une valeur de 396, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle.

True Negatives (TN) : Avec une valeur de 11 413, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle SVM.

### e) Arbre de Décision (Decision Tree)

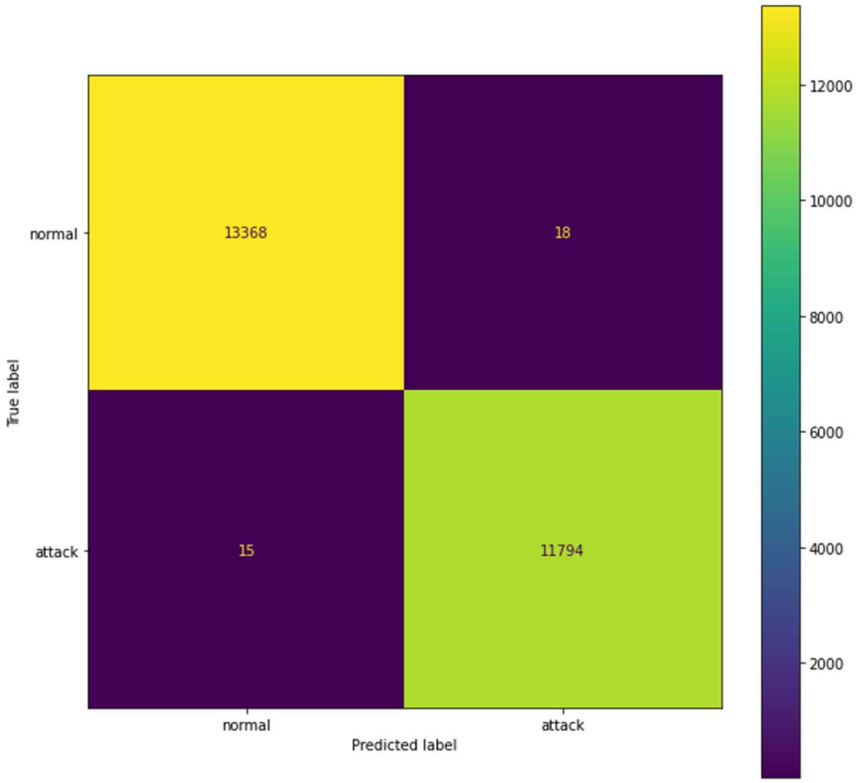


Figure 3. 6 : Matrice de Confusion Arbre de Décision (Decision Tree) pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de l'arbre de Décision pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 368, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle Arbre de Décision.

False Positives (FP) : Avec une valeur de 18, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle.

False Negatives (FN) : Avec une valeur de 15, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle.

True Negatives (TN) : Avec une valeur de 11 794, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle Arbre de Décision.

### f) Random forest

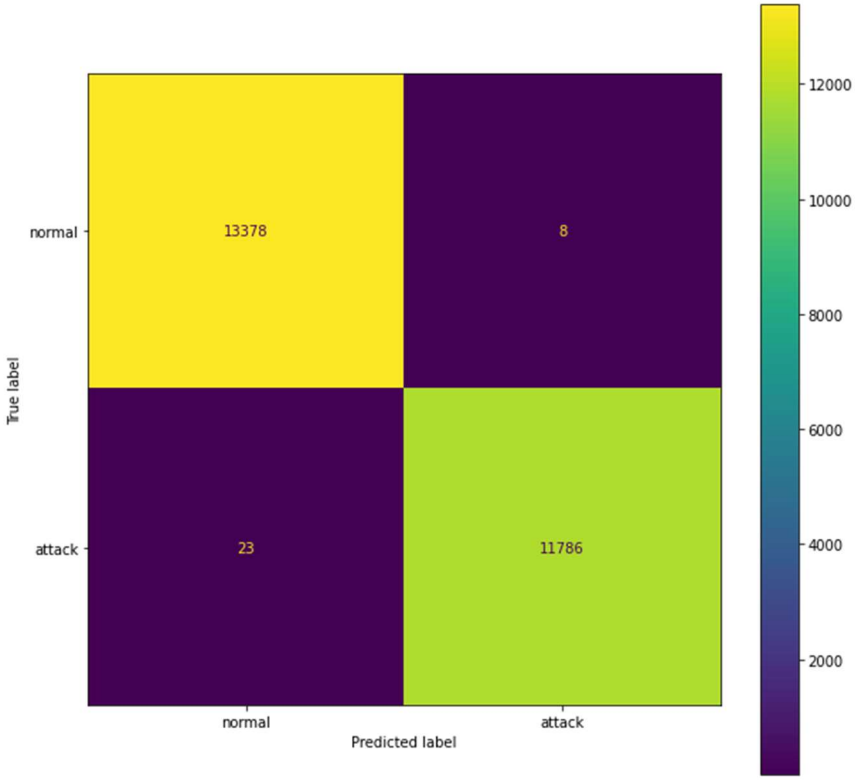


Figure 3. 7 : Matrice de Confusion Random forest pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de la Random forest pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 378, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle Random Forest.

False Positives (FP) : Avec une valeur de 8, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle.

False Negatives (FN) : Avec une valeur de 23, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle.

True Negatives (TN) : Avec une valeur de 11 786, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle Random Forest.

### g) Modèle Random Forest après l'application de PCA (Principal Component Analysis)

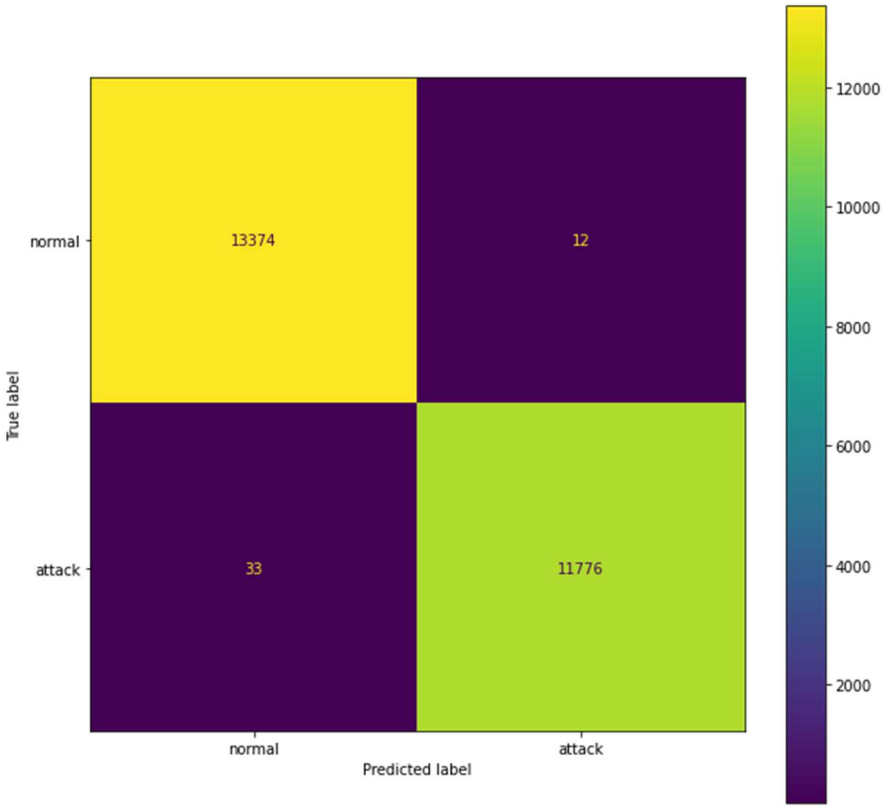


Figure 3. 8 : Matrice de Confusion Modèle Random Forest après l'application de PCA pour la Classification (Attack, Normal).

**Explication Matrice de Confusion de la Modèle Random Forest après l'application de PCA pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 374, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles après l'application de PCA par le modèle Random Forest.

False Positives (FP) : Avec une valeur de 12, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" après l'application de PCA par le modèle Random Forest.

False Negatives (FN) : Avec une valeur de 33, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" après l'application de PCA par le modèle Random Forest.

True Negatives (TN) : Avec une valeur de 11 776, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles après l'application de PCA par le modèle Random Forest.

### h) XGBoost

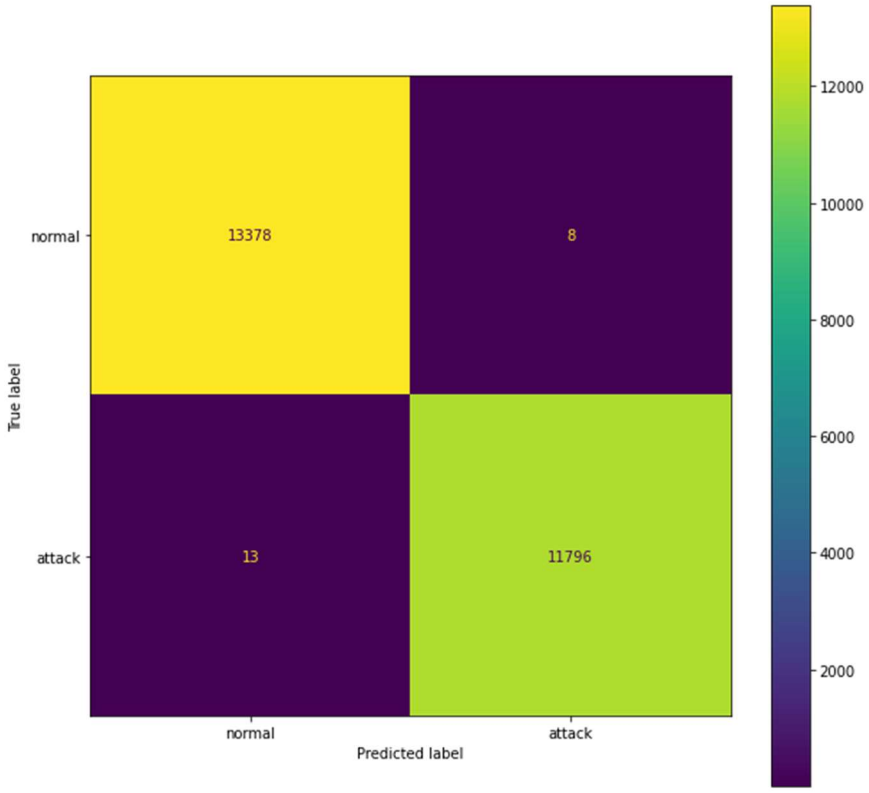


Figure 3. 9 : Matrice de Confusion XGBoost pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de XGBoost pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 378, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle XGBoost.

False Positives (FP) : Avec une valeur de 8, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle XGBoost.

False Negatives (FN) : Avec une valeur de 13, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle XGBoost.

True Negatives (TN) : Avec une valeur de 11 796, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle XGBoost.

### i) RNA (Réseau de Neurones Artificiels)

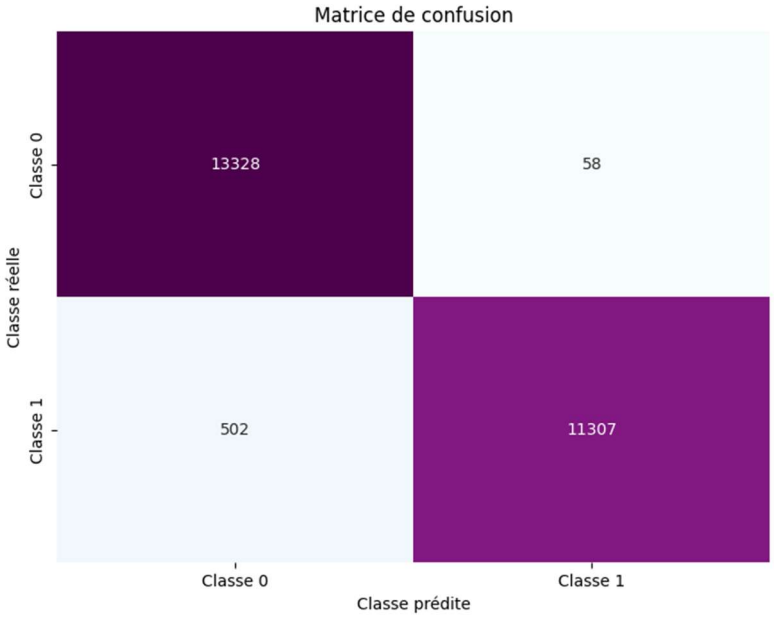


Figure 3. 10 : la matrice de confusion du modèle RNA pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de RNA pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 328, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle RNA.

False Positives (FP) : Avec une valeur de 58, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle RNA.

False Negatives (FN) : Avec une valeur de 502, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle RNA.

True Negatives (TN) : Avec une valeur de 11 307, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle RNA.

### j) CNN (réseau de neurones convolutifs)

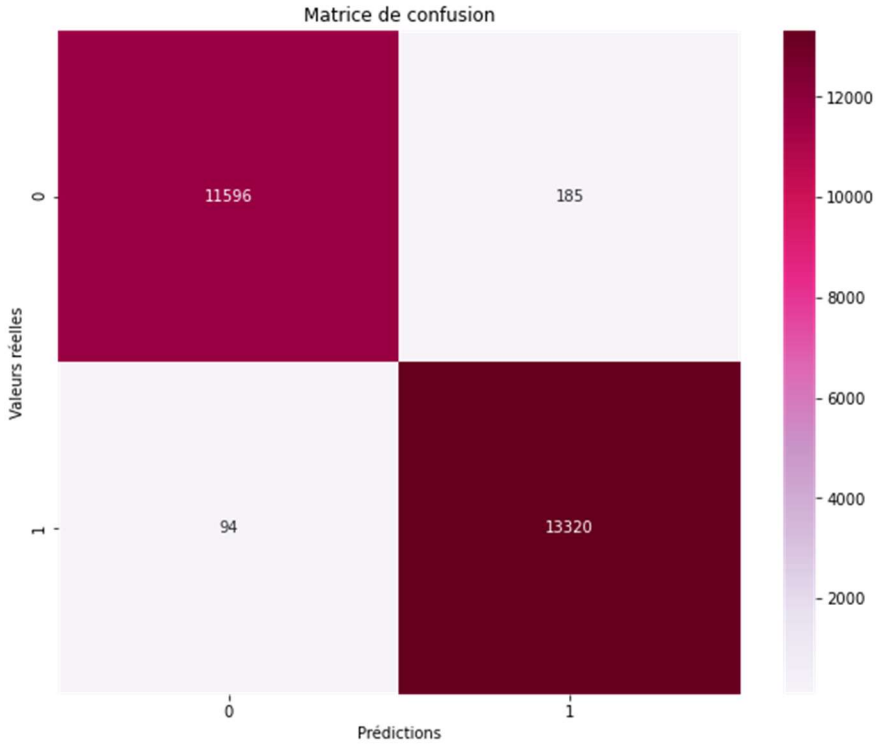


Figure 3. 11: la matrice de confusion du modèle CNN pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de CNN pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 11 596, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle CNN.

False Positives (FP) : Avec une valeur de 185, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle CNN.

False Negatives (FN) : Avec une valeur de 94, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle CNN.

True Negatives (TN) : Avec une valeur de 13 320, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle CNN.

### k) RNN (Réseau de Neurones Récurrents)

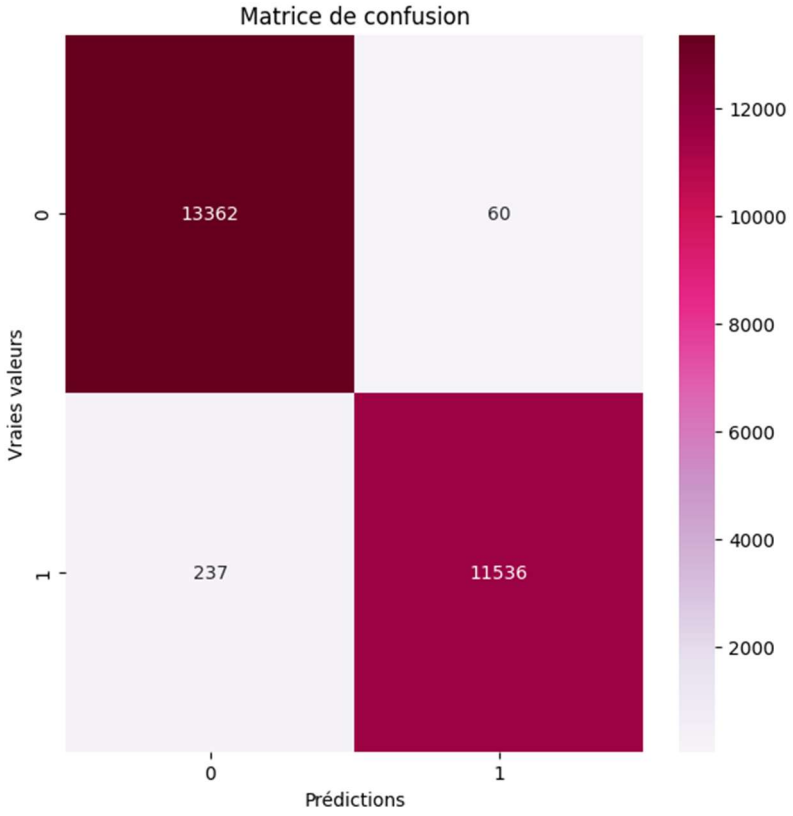


Figure 3. 12: la matrice de confusion du modèle RNN pour la Classification (Attack, Normal)

**Explication Matrice de Confusion de RNN pour la Classification (Attack, Normal)**

True Positives (TP) : Avec une valeur de 13 362, cela représente le nombre d'observations qui appartiennent à la classe "attack" et qui ont été correctement prédites comme telles par le modèle RNN.

False Positives (FP) : Avec une valeur de 60, cela représente le nombre d'observations qui appartiennent à la classe "normal" mais qui ont été incorrectement prédites comme "attack" par le modèle RNN.

False Negatives (FN) : Avec une valeur de 237, cela représente le nombre d'observations qui appartiennent à la classe "attack" mais qui ont été incorrectement prédites comme "normal" par le modèle RNN.

True Negatives (TN) : Avec une valeur de 11 536, cela représente le nombre d'observations qui appartiennent à la classe "normal" et qui ont été correctement prédites comme telles par le modèle RNN.

### 3.2.4 les paramètres de chaque modèle utilisé

Le tableau fournit une liste de modèles d'apprentissage automatique et leurs paramètres respectifs. Ces modèles sont largement utilisés pour résoudre des problèmes de classification et de prédiction.

| Modèle | Paramètres du modèle |
|---|---|
| RandomForest | bootstrap=True, ccp_alpha=0.0, criterion='gini', max_features='sqrt', n_estimators=100 |
| Logistic Regression | C=1.0, max_iter=100, solver='lbfgs', penalty='l2' |
| KNeighborsClassifier | algorithm='auto', n_neighbors=5, weights='uniform' |
| DecisionTreeClassifier | ccp_alpha=0.0, criterion='gini', min_samples_leaf=1, min_samples_split=2 |
| CNN | Conv1D: Filters=128, Kernel=2, Activation='ReLU', MaxPooling1D: Window=2, Dropout: Rate=0.2, Conv1D: Filters=256, Kernel=2, Activation='ReLU', MaxPooling1D: Window=2, Dropout: Rate=0.4, Dense: Neurons=128, Activation='ReLU', Dropout: Rate=0.4, Dense (output): Neurons=1, Activation='Sigmoid' |
| XGBoost | objective='binary:logistic', n_estimators=100 |
| SVM | C=1.0, kernel='rbf' |

| Modèle | Paramètres du modèle |
|---|---|
| Naive Bayes | var_smoothing=1e-09 |
| RNA | Activation='ReLU', Alpha=0.0001, Hidden Layers= (100,), Learning Rate=0.001, Solver='Adam', Tolerance=0.0001, Validation Fraction=0.1 |
| RNN | Batch Input Shape= (None, 10, 1), Units=32, Activation='Tanh' |

Tableau 3. 2 : Modèles d'apprentissage automatique et leurs paramètres associés.

RandomForest : Cet algorithme est basé sur des arbres de décision et utilise une technique d'échantillonnage avec remplacement appelée bootstrap. Il utilise également le critère de Gini pour diviser les nœuds de l'arbre. Le modèle est composé de 100 estimateurs.

Logistic Regression : Il s'agit d'un modèle de régression logistique qui est couramment utilisé pour la classification. Il utilise le solveur lbfgs pour optimiser la fonction de coût et applique une régularisation de type l2 avec un paramètre C de 1.0. Le modèle effectue un maximum de 100 itérations.

KNeighborsClassifier : Ce modèle de classification se base sur les k plus proches voisins pour prendre des décisions. Il utilise l'algorithme auto pour la recherche des voisins et attribue des poids uniformes à chaque voisin. Le modèle considère les 5 voisins les plus proches.

DecisionTreeClassifier : Ce modèle utilise un arbre de décision pour effectuer des classifications. Il utilise le critère de Gini pour mesurer la qualité des divisions de nœuds. Le modèle ne nécessite qu'un seul échantillon pour créer une feuille et effectue des divisions même avec un seul échantillon.

CNN (Convolutional Neural Network) : Ce modèle de réseau neuronal convolutif utilise deux couches de convolution avec des filtres de taille 128 et 256, suivies de couches de MaxPooling1D pour réduire la dimensionnalité. Des couches denses avec des fonctions d'activation ReLU sont utilisées pour la classification binaire.

XGBoost : Il s'agit d'un algorithme de boosting qui est efficace pour les problèmes de classification. Il utilise une fonction d'objectif de classification binaire avec une logistique. Le modèle est composé de 100 estimateurs.

SVM (Support Vector Machine) : Ce modèle est largement utilisé pour la classification binaire. Il utilise le noyau rbf (radial basis function) et un paramètre de régularisation C de 1.0.

Naive Bayes : Ce modèle probabiliste est basé sur le théorème de Bayes. Il utilise un paramètre de lissage (var_smoothing) de 1e-09 pour éviter les problèmes de division par zéro.

RNA (Réseau de Neurones Artificiels) : Ce modèle de réseau neuronal utilise une activation ReLU et un solveur Adam pour optimiser la fonction de coût. Il comprend une couche cachée avec 100 neurones et un taux d'apprentissage de 0.001.

RNN (Réseau de Neurones à Mémoire Récurrente) : Ce modèle de réseau neuronal récurrent est adapté aux données séquentielles. Il utilise une activation Tanh et dispose de 32 unités pour capturer les dépendances à long terme.

Ces modèles et leurs paramètres offrent une flexibilité pour résoudre divers problèmes d'apprentissage automatique. En fonction du contexte et des caractéristiques des données, il est possible de choisir le modèle et les paramètres les mieux adaptés pour obtenir de bons résultats.

## 3.3 Évaluation des performances des modèles

### 3.3.1 Évaluation des performances des algorithmes de Machine Learning

En ce qui concerne l'exactitude d'entraînement, les résultats les plus performants sont obtenus avec RandomForestClassifier (99,994%) et DecisionTreeClassifier (99,994%), qui présentent tous deux des scores très élevés. Ces deux algorithmes démontrent une capacité élevée à apprendre les motifs et à classer correctement les exemples d'entraînement.

En ce qui concerne l'exactitude sur l'ensemble de test, RandomForestClassifier (99,881%) se distingue comme le meilleur algorithme, suivi de près par XGBoost (99.991%) et le réseau de neurones convolutionnels (99,412%). Ces algorithmes ont démontré leur capacité à généraliser et à classer correctement les exemples inconnus.

En ce qui concerne la précision, KNN (KNeighborsClassifier) se distingue à la fois sur l'ensemble d'entraînement (99,225%) et sur l'ensemble de test (99,056%). Il présente une bonne capacité à classifier correctement les échantillons positifs. RandomForestClassifier (99,941%) et XGBoost (99.916%) obtiennent également de bons résultats en termes de précision.

En ce qui concerne le rappel, RandomForestClassifier est l'algorithme le plus performant à la fois sur l'ensemble d'entraînement, avec un bon résultat de (99,994%), et sur l'ensemble de test (99,805%). Il est capable d'identifier la plupart des échantillons positifs. Les autres algorithmes affichent également de bons résultats en termes de rappel.

En résumé, RandomForestClassifier se distingue dans la plupart des métriques, avec de bons résultats en termes d'exactitude, de précision et de rappel, tant sur l'ensemble d'entraînement que sur l'ensemble de test. DecisionTreeClassifier et XGBoost affichent également des performances solides dans plusieurs métriques.

### 3.3.2 Caractéristiques distinctes des algorithmes de Machine Learning

1) **Complexité des modèles**

- RandomForestClassifier : La complexité dépend du nombre d'arbres de décision dans la forêt.
- SVM (Linear SVC) : La complexité dépend de la dimensionnalité des données et du nombre d'échantillons.
- Logistic Regression : La complexité est faible et linéaire par rapport au nombre de caractéristiques.
- KNN (KNeighborsClassifier) : La complexité est généralement élevée en raison du calcul des distances.
- DecisionTreeClassifier : La complexité dépend de la profondeur de l'arbre et du nombre de caractéristiques.
- Naive Bayes : La complexité est généralement faible grâce à l'hypothèse d'indépendance conditionnelle.
- Réseau de neurones artificiels : La complexité dépend de l'architecture du réseau.
- XGBoost : La complexité dépend du nombre d'arbres de décision utilisés.
- Réseau de neurones convolutionnels : La complexité dépend de l'architecture du réseau, y compris le nombre de couches de convolution et de filtres.
- Réseau de neurones récurrents : La complexité dépend de l'architecture du réseau, y compris le nombre de couches récurrentes et de neurones.

2) **Capacité à traiter les données en temps réel**

- RandomForestClassifier : Peut traiter les données en temps réel, mais la prédiction peut prendre plus de temps avec de nombreux arbres.
- SVM (Linear SVC) : Peut traiter les données en temps réel, mais peut être lent pour de grandes dimensions ou de grands ensembles de données.
- Logistic Regression : Peut traiter les données en temps réel de manière efficace.
- KNN (KNeighborsClassifier) : Peut-être plus lent pour les prédictions en temps réel en raison du calcul des distances.

- DecisionTreeClassifier : Peut traiter les données en temps réel de manière efficace.
- Naive Bayes : Peut traiter les données en temps réel de manière efficace.
- Réseau de neurones artificiels : Peut traiter les données en temps réel, mais la vitesse peut varier en fonction de l'architecture du réseau.
- XGBoost : Peut traiter les données en temps réel de manière efficace.
- Réseau de neurones convolutionnels : Peut traiter les données en temps réel, mais la vitesse peut varier en fonction de l'architecture du réseau.
- Réseau de neurones récurrents : Peut traiter les données en temps réel, mais la vitesse peut varier en fonction de l'architecture du réseau.

**3) Facilité d'implémentation**

- RandomForestClassifier : Relativement facile à implémenter avec des bibliothèques comme scikit-learn.
- SVM (Linear SVC) : Facile à implémenter avec des bibliothèques comme scikit-learn.
- Logistic Regression : Facile à implémenter avec des bibliothèques comme scikit-learn.
- KNN (KNeighborsClassifier) : Facile à implémenter avec des bibliothèques comme scikit-learn.
- DecisionTreeClassifier : Facile à implémenter avec des bibliothèques comme scikit-learn.
- Naive Bayes : Facile à implémenter avec des bibliothèques comme scikit-learn.
- Réseau de neurones artificiels : Peut nécessiter une connaissance avancée des réseaux neuronaux et une mise en œuvre personnalisée.
- XGBoost : Peut nécessiter une configuration avancée, mais dispose de bibliothèques pour faciliter l'implémentation.
- Réseau de neurones convolutionnels : Peut nécessiter une connaissance avancée des réseaux neuronaux et une mise en œuvre personnalisée.
- Réseau de neurones récurrents : Peut nécessiter une connaissance avancée des réseaux neuronaux et une mise en œuvre personnalisée.

**4) Robustesse aux données manquantes ou bruitées**

- RandomForestClassifier : Relativement robuste aux données manquantes ou bruitées.
- SVM (Linear SVC) : Moins robuste aux données manquantes ou bruitées, nécessite un prétraitement supplémentaire.
- Logistic Regression : Moins robuste aux données manquantes ou bruitées, nécessite un prétraitement supplémentaire.
- KNN (KNeighborsClassifier) : Moins robuste aux données manquantes ou bruitées, nécessite un prétraitement supplémentaire.

- DecisionTreeClassifier : Relativement robuste aux données manquantes ou bruitées.
- Naive Bayes : Relativement robuste aux données manquantes ou bruitées.
- Réseau de neurones artificiels : Peut être sensible aux données manquantes ou bruitées, nécessite souvent un prétraitement supplémentaire.
- XGBoost : Relativement robuste aux données manquantes ou bruitées.
- Réseau de neurones convolutionnels : Peut être sensible aux données manquantes ou bruitées, nécessite souvent un prétraitement supplémentaire.
- Réseau de neurones récurrents : Peut être sensible aux données manquantes ou bruitées, nécessite souvent un prétraitement supplémentaire.

**5) Interprétabilité des modèles**

- RandomForestClassifier : Moins interprétable en raison de l'ensemble d'arbres et de leurs interactions complexes.
- SVM (Linear SVC) : Relativement interprétable, en particulier avec un noyau linéaire.
- Logistic Regression : Relativement interprétables, les coefficients peuvent être interprétés pour évaluer l'importance des caractéristiques.
- KNN (KNeighborsClassifier) : Moins interprétable en raison de sa nature basée sur les voisins.
- DecisionTreeClassifier : Relativement interprétables, les règles de décision de l'arbre peuvent être facilement comprises.
- Naive Bayes : Relativement interprétable, en particulier en ce qui concerne les probabilités conditionnelles.
- Réseau de neurones artificiels : Moins interprétable en raison de la complexité des connexions entre les neurones.
- XGBoost : Moins interprétable en raison de l'ensemble d'arbres et de leurs interactions complexes.
- Réseau de neurones convolutionnels : Moins interprétable en raison de la complexité des connexions entre les neurones et des opérations de convolution.
- Réseau de neurones récurrents : Moins interprétable en raison de la complexité des connexions récurrentes.

## 3.4 Comparaison des résultats obtenus

### 3.4.1 Analyse Comparative des Performances des Algorithmes

En comparant les métriques de performances des différents algorithmes sur la base de données fournie, on peut observer les résultats suivants :

En termes d'exactitude d'entraînement, RandomForestClassifier et DecisionTreeClassifier obtiennent tous deux des scores très élevés.

En ce qui concerne l'exactitude sur l'ensemble de tests, RandomForestClassifier est l'algorithme le plus performant, suivi de près par XGBoost et le réseau de neurones convolutionnels.

En ce qui concerne la précision, KNN (KNeighborsClassifier) se démarque tant sur l'ensemble d'entraînement que sur l'ensemble de tests. RandomForestClassifier et XGBoost obtiennent également de bons résultats en termes de précision.

En ce qui concerne le rappel, RandomForestClassifier est l'algorithme le plus performant à la fois sur l'ensemble d'entraînement et sur l'ensemble de tests.

Sur la base de ces résultats, RandomForestClassifier semble être le meilleur choix, car il obtient de bons résultats en termes d'exactitude, de précision et de rappel, tant sur l'ensemble d'entraînement que sur l'ensemble de tests. Cependant, il est important de prendre en compte d'autres critères spécifiques à notre étude, telle que la complexité du modèle, la capacité à traiter les données en temps réel, la facilité d'implémentation, la robustesse aux données manquantes ou bruitées, ainsi que l'interprétabilité des modèles, afin de faire un choix éclairé en fonction de nos besoins et contraintes spécifiques.

### 3.4.2 Analyse Comparative des Algorithmes : Complexité, Temps Réel, Implémentation, Robustesse et Interprétabilité

**Complexité des modèles :** L'algorithme présentant la complexité la plus faible est la régression logistique (Logistic Regression), suivie du Naive Bayes. Cependant, cela dépend également de la taille et de la nature spécifique de votre ensemble de données. Il est important de considérer la relation entre la complexité du modèle et les performances globales.

**Capacité à traiter les données en temps réel :** Les algorithmes ayant une meilleure capacité à traiter les données en temps réel sont la régression logistique (Logistic Regression), l'arbre de décision (DecisionTreeClassifier), le Naive Bayes et le XGBoost. Ils sont plus rapides pour effectuer des prédictions en temps réel, bien que cela puisse varier en fonction de la complexité de l'ensemble de données et de

l'architecture du modèle.

**Facilité d'implémentation :** Les algorithmes les plus faciles à mettre en œuvre sont le RandomForestClassifier, la SVM linéaire (Linear SVC), la régression logistique (Logistic Regression), le KNN (KNeighborsClassifier), l'arbre de décision (DecisionTreeClassifier) et le Naive Bayes, car ils disposent d'implémentations disponibles dans des bibliothèques populaires telles que scikit-learn.

**Robustesse aux données manquantes ou bruyantes :** Les algorithmes les plus robustes aux données manquantes ou bruyantes sont le RandomForestClassifier, l'arbre de décision (DecisionTreeClassifier), le Naive Bayes et le XGBoost. Ils sont généralement moins sensibles aux valeurs manquantes ou aux données bruitées.

**Interprétabilité des modèles :** Les algorithmes les plus interprétables sont la régression logistique (Logistic Regression), la SVM linéaire (Linear SVC), l'arbre de décision (DecisionTreeClassifier) et le Naive Bayes. Ces modèles permettent une compréhension plus facile des résultats grâce à leurs règles de décision ou à l'interprétation des coefficients.

- ✓ Il est important de prendre en compte tous ces facteurs et de les pondérer en fonction de vos besoins spécifiques avant de choisir un algorithme

Si vous recherchez une complexité de modèle faible, la régression logistique et le Naive Bayes sont de bons choix. Cependant, cela dépendra également de la taille et de la nature spécifique de votre ensemble de données.

En ce qui concerne la capacité à traiter les données en temps réel, la régression logistique, l'arbre de décision, le Naive Bayes et le XGBoost sont souvent considérés comme rapides pour effectuer des prédictions en temps réel. Cependant, la vitesse peut varier en fonction de la complexité de l'ensemble de données et de l'architecture du modèle.

En termes de facilité d'implémentation, le RandomForestClassifier, la SVM linéaire, la régression logistique, le KNN, l'arbre de décision et le Naive Bayes sont généralement considérés comme faciles à mettre en œuvre, car ils disposent d'implémentations disponibles dans des bibliothèques populaires telles que scikit-learn.

Pour ce qui est de la robustesse aux données manquantes ou bruyantes, le RandomForestClassifier, l'arbre de décision, le Naive Bayes et le XGBoost sont souvent plus robustes, car ils sont moins sensibles aux valeurs manquantes ou aux données bruitées.

Enfin, en termes d'interprétabilité des modèles, la régression logistique, la SVM linéaire, l'arbre de décision et le Naive Bayes sont souvent considérés comme plus interprétables en raison de leurs règles de décision ou de l'interprétation des coefficients.

### 3.4.3 Analyse Comparative F-mesure (F1-score)

F-mesure (F1-score) : La F-mesure est une mesure combinée de la précision et du rappel, donnant une indication globale de la performance du modèle. Elle est calculée comme une moyenne pondérée de la précision et du rappel.

Le score F1 peut être calculé à l'aide de la formule suivante :

$$\mathbf{F1} = \mathbf{2} \times \frac{(\mathbf{Precision} \times \mathbf{Recall})}{(\mathbf{Precision} + \mathbf{Recall})} \tag{3.4}$$

| Algorithme | F1 score |
|---|---|
| RandomForestClassifier | 0.99873 |
| SVM (Linear SVC) | 0.96492 |
| Logistic Regression | 0.86684 |
| KNN (KNeighborsClassifier) | 0.98863 |
| DecisionTreeClassifier | 0.99869 |
| Naive Bayes | 0.90981 |
| Réseau de neurones artificiels | 0.97533 |
| XGBoost | 0.99911 |
| Réseau de neurones convolutionnels | 0.99411 |
| Réseau de neurones récurrents | 0.99239 |

Tableau 3. 3 : Résultat de F1score.

**Commentaire :**

En analysant les scores F1 :

Les algorithmes RandomForestClassifier, DecisionTreeClassifier, XGBoost et le réseau de neurones convolutionnels ont obtenu les meilleurs scores F1, dépassant les 0,99. Cela indique que ces modèles ont des performances exceptionnelles en termes de précision et de rappel, et sont capables de bien classifier les exemples positifs tout en minimisant les faux positifs et les faux négatifs.

Les algorithmes SVM (Linear SVC), KNN (KNeighborsClassifier), et le réseau de neurones artificiels ont également obtenu de bons scores F1, se situant autour de 0,96 à 0,99. Ces modèles parviennent à bien équilibrer la précision et le rappel dans leurs prédictions.

Les algorithmes Logistic Regression, Naive Bayes et le réseau de neurones récurrents ont obtenu des scores F1 légèrement inférieurs, se situant autour de 0,86 à 0,92. Bien que ces scores soient légèrement inférieurs à certains autres algorithmes, ils restent dans une plage raisonnable et indiquent que ces modèles sont capables de classifier avec une précision raisonnable tout en maintenant un équilibre acceptable entre la précision et le rappel.

## 3.5 Conclusion

En conclusion, XGBoost est recommandé pour la détection d'intrusions réseau en raison de sa précision élevée, de sa capacité à minimiser les erreurs de classification et de sa bonne performance en termes de rappel. Cependant, les performances peuvent varier selon l'environnement et les données, nécessitant des expérimentations supplémentaires. Globalement, ce modèle constitue une solution prometteuse, mais d'autres approches et évaluations sont nécessaires pour garantir une détection fiable et efficace des intrusions réseau.

# Conclusion générale

# Conclusion générale

La présente étude s'est concentrée sur l'utilisation du machine learning et du deep learning, pour la détection d'intrusions réseau. Nous avons exploré les concepts de base de la détection d'intrusions, les techniques traditionnelles utilisées jusqu'à présent, ainsi que les principes fondamentaux du machine learning appliqués à ce domaine spécifique.

À travers une inspection de la littérature, nous avons examiné les avancées récentes dans le domaine de la détection d'intrusions en utilisant des approches basées sur le machine learning. Cette analyse nous a permis d'identifier les tendances actuelles, les méthodologies utilisées et les résultats obtenus. Nous avons constaté que les techniques de deep learning, telles que les réseaux de neurones profonds, ont montré des performances prometteuses en matière de détection d'intrusions, permettant de capturer des caractéristiques complexes et de détecter des attaques insidieuses avec une précision accrue.

En utilisant différentes techniques de machine learning, notamment KNN, SVM, Decision tree, Random Forest, CNN, RNN, régression logistique, xgboost et naive bayes, nous avons évalué les performances de ces modèles dans la détection d'intrusions réseau. Les résultats obtenus ont démontré que certaines techniques de deep learning, telles que les réseaux de neurones profonds, ont surpassé les approches traditionnelles en termes de précision et de rappel.

En synthétisant les résultats de cette étude, nous pouvons conclure que l'utilisation du machine learning et du deep learning, présente un potentiel significatif pour améliorer la détection d'intrusions réseau. Les modèles de machine learning ont montré leur capacité à identifier des schémas et des comportements anormaux, permettant une détection

précoce des attaques et une meilleure protection des réseaux.

Nos contributions dans ce projet résident dans l'évaluation et la comparaison des différentes techniques de machine learning pour la détection d'intrusions réseau. Nous avons fourni des informations utiles sur les performances des modèles et leurs avantages respectifs. De plus, notre revue de la littérature a permis d'identifier les lacunes actuelles dans le domaine de la détection d'intrusions et d'ouvrir des perspectives pour des travaux futurs.

Pour les travaux futurs, il convient d'explorer davantage les techniques de deep learning, en utilisant des architectures plus avancées de réseaux de neurones, telles que les réseaux adversaires génératifs (GAN), pour améliorer la détection des attaques sophistiquées. De plus, il est intéressant de se pencher sur l'intégration de l'intelligence artificielle, notamment de l'apprentissage par renforcement, pour développer des systèmes de détection d'intrusions plus adaptatifs et capables de s'auto-améliorer en réponse aux nouvelles attaques. La détection d'intrusions réseau en utilisant le Machine Learning est une approche prometteuse pour renforcer la sécurité des réseaux informatiques. En développant une application mobile dédiée à cette tâche, les utilisateurs pourront bénéficier d'une surveillance en temps réel et de notifications d'intrusions potentielles. Ce projet offre un potentiel significatif pour améliorer la sécurité des réseaux et protéger les utilisateurs contre les menaces actuelles et futures.

En conclusion, cette étude a démontré l'efficacité et le potentiel du machine learning, et le deep learning, dans la détection d'intrusions réseau. Les résultats obtenus et les perspectives identifiées ouvrent la voie à de nouvelles avancées dans le domaine de la sécurité des réseaux, contribuant ainsi à la protection des systèmes informatiques contre les menaces croissantes des attaques

# Bibliographie

[1] A. Jaiswal, S. K. Dhal, et S. H. Naqvi, « Network Intrusion Detection using Deep Learning: A Review », International Journal of Engineering and Advanced Technology (IJEAT), vol. 8, no 4, Avril 2019.

[2] M. N. Saad et F. A. Hussain, « A Comparative Study of Machine Learning Techniques for Network Intrusion Detection », in 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 2019.

[3] I. Yildiz, E. Avcı, et B. Tuncel, « Network Intrusion Detection with Deep Learning Approaches: A Review », in 2021 IEEE 7th International Conference on Computer and Communications (ICCC), Chengdu, China, 2021.

[4] Geekflare. (s.d.). IDS vs IPS - Network Security Solutions Dans Geekflare. Récupéré le 1 mai 2023, de https://geekflare.com/fr/ids-vs-ips-network-security-solutions/

[5] Abou Haidar, Gaby et Boustany, Charbel. "High Perception Intrusion Detection Systems Using Neural Networks." Dans les actes du 2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), Beyrouth, Liban, pp. 2-3, 2015.

[6] Tawbi, N., Mahdaoui, S., & Chouarfia, A. (2021). Threat intelligence sharing platform architecture for the cloud. Journal of Cybersecurity and Privacy, 1(1), Article 10. Consulté le 2 mai 2023, de https://cybersecurity.springeropen.com/articles/10.1186/s42400-021-00077-7

[7] Tsai, C.-F., Hsu, Y.-F., Lin, C.-Y., & Lin, W.-Y. (2009). Intrusion detection by machine learning: A review. Expert Systems with Applications, 36(10), 2-5. Taipei, Taiwan.

[8] Haripriya, L., & Jabbar, M.A. (2018). Role of Machine Learning in Intrusion Detection System: Review. Dans Proceedings of the 2nd International conference on Electronics, Communication and Aerospace Technology (ICECA 2018) (p. numéro de page). IEEE Conference Record # 42487 ; IEEE Xplore ISBN : 978-1-5386-0965-1.

[9] Karatas, G., Demir, O., & Sahingoz, O.K. (2018). Deep Learning in Intrusion Detection Systems. Dans Proceedings of the International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (p. 1-3). Ankara, Turkey.

[10] IEEE. (2023, 28 mars). Nguyen, D.T., Armitage, G., & Vilas, M. (2018). A Survey of machine learning for big data processing in intrusion detection systems. Journal of Network and Computer Applications, 115, 18-31.
[11] Mirjalili, S., Mirjalili, S.M., & Lewis, A. (2021). Intrusion detection using machine learning techniques: A comprehensive review. Computers & Security, 96, 102206.
[12] Idowu, A.I., Rosado, D.G., & Shiaeles, S. (2020). Anomaly-based intrusion detection using machine learning techniques: A review. Computers & Security, 94, 101894.
[13] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A.A. (2019). Deep learning-based intrusion detection systems: A review. ACM Computing Surveys, 51(3), 53.
[14] J. Smith, "Python for Data Analysis," O'Reilly Media, New York, 2020.
[15] Kaur, H., & Singh, A. (2021). A comprehensive analysis of NSL-KDD dataset for intrusion detection system using machine learning techniques. International Journal of Advanced Science and Technology, 30(5), 1512-1522.

[16] Raza, M., Rasheed, M. A., Khan, M. A., & Shahzad, M. (2022). A comprehensive survey of data preprocessing techniques in machine learning. Computers, Materials & Continua, 71(1), 1211-1234.

[17] Liu, Y., Yu, D., Huang, Y., & Chen, C. (2022). Feature selection for improving machine learning models: An analysis of NSL-KDD data. Journal of Systems and Software, 185, 111464.

[18] Saeed, K., Lee, S., Kim, J., & Kim, H. J. (2022). Network traffic classification using machine learning: A comprehensive survey. IEEE Communications Surveys & Tutorials, 24(2), 1334-1380.

[19] Datatron. (2023). What is a support vector machine? [Figure] Consulté le 27 mai 2023, de https://datatron.com/what-is-a-support-vector-machine/

[20] ResearchGate. (2023). Illustration of random forest trees [Figure] Consulté le 26 mai 2023, de https://www.researchgate.net/figure/Illustration-of-random-forest-trees_fig4_354354484.

[21] BuiltIn. (2023) Logistic Regression: A Classification Algorithm. Builtin.com. Consulté le 25 mai 2023, de https://builtin.com/machine-learning/logistic-regression-classification-algorithm

[22] DataScientest. (2023). KNN. DataScientest.com. Consulté le 30 mai 2023, de https://datascientest.com/knn

[23] JavaTpoint. (2023). JavaTpoint - A Solution of all Technology. Consulté le 31 mai 2023, à partir de https://www.javatpoint.com/

[24] Haykin, S. (1999). Neural Networks: A Comprehensive Foundation. Prentice Hall.

[25] Li, J., Wei, Y., Du, N., & Wei, W. (2020). A Comprehensive Review of Recurrent Neural Network Architectures.

[26] Shehab, M., Badr, N. A., Nabil, A., & Elhoseny, H. M. (2021). Deep Convolutional Neural Networks for Network Intrusion Detection: An Overview and a Proposal. IEEE Access.

[27] Exxact Corporation. (2023). Image Classification with DCNNs [Figure]. Consulté de https://www.exxactcorp.com/blog/Deep-Learning/Image-Classification-with-DCNNs

[28] Chollet, F. (2017). Deep learning with Python. Manning Publications.

[29] Powers, D. M. (2021). Evaluation: Precision, Recall, F-Measure, and Matthews Correlation Coefficient. In Encyclopedia of Bioinformatics and Computational Biology (pp. 822-829). Elsevier.