République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A. MIRA-BEJAIA
Faculté des Sciences Exactes
Département d'Informatique

Mémoire de fin de cycle

En vue de l'obtention du diplôme de master de recherche en informatique
Option : Réseau et sécurité

Thème

Implémentation d'un IDS basé sur des approches de machine et deep learning pour la détection d'attaques de botnets dans l'iot

Réalisé par :

Mlle ABBACHE Imene
Mlle AGUEMAL Karima

Soutenu le 21 juin 2023, Devant le jury composé de :

AMROUN Kamal	Professeur	Encadrant	Univ. de Béjaia
EL-SAKAAN Nadim	Doctorant	Co-Encadrant	Univ. de Béjaia
MOKTEFI Mohand	M.C.B	Président	Univ. de Béjaia
SABRI Salima	M.C.B	Examinatrice	Univ. de Béjaia

Année Universitaire : 2022/2023

Dédicaces

Je dédie ce travail, à moi même pour mes efforts et ma patience.
À mes chers parents, qui ont toujours cru en moi et m'ont encouragé à poursuivre mes rêves.
À ma sœur, qui a toujours été là pour moi, à toute ma famille, qui m'a apporté son soutien,.
À mon encadrant, qui m'a guidé avec patience et bienveillance tout au long de ce projet.
À mon amie et ma binôme Karima.
À tous ceux qui ont contribué à la réussite de ce travail.

Imene

C'est avec un grand plaisir que je dédie ce travail à mes chers parents, source de vie et d'amour pour leur soutien, leur encouragement et leur patience durant mon parcours scolaire.
À mes frères et ma sœur ainsi qu'à ma famille, mes amis et à tous ceux qui ont participé à ma réussite.
À Imene, chère amie avant d'être ma binôme.
À Amayas, mon ami pour son aide précieuse.
À vous chers lecteurs.

Karima

Remerciements

Nous remercions en premier lieu DIEU pour nous avoir donné la force de réaliser ce travail.

Nous tenons à exprimer nos sincères remerciements à toutes les personnes qui ont contribué à la réalisation de ce travail.

Nous voudrions tout d'abord remercier notre directeur de mémoire, M. EL-SAKAAN Nadim, pour son aide précieuse, sa patience, et ses conseils tout au long de ce projet. Sa disponibilité, ses remarques pertinentes et sa rigueur scientifique nous ont permis de mener à bien cette recherche.

Nous tenons également à remercier nos parents pour leur soutien indéfectible, leur amour et leur confiance en nous. Leur présence a été un moteur pour nous tout au long de nos études.

Nous souhaitons aussi adresser nos remerciements à nos frères et sœurs, pour leur soutien moral et leur présence lors des moments difficiles, ainsi qu'à toute la famille pour leur soutien inconditionnel.

Nous remercions également tous nos amis qui ont contribué à leur manière à la réalisation de ce travail.

Enfin, nous souhaitons remercier les membres du jury M. MOKTEFI Mohand et Mme. SABRI Salima d'avoir accepté d'évaluer notre travail.

Nous remercions aussi l'ensemble des enseignants et des membres de l'administration de l'université qui ont participé à notre formation et nous ont permis d'acquérir les connaissances nécessaires pour mener à bien ce travail.

Merci à tous.

Imene et Karima

Table des matières

Liste des tableaux

Table des figures

Liste des abréviations

ANN Artificial Neural Network
CNN (Convolutional Neural Network
DDos Distributed denial-of-service
Dos Denial-of-service
DeepL Deep Learning
DNN deep neural network
DT Decision Tree
FN False Negatif
FP False Positif
GNB Gaussian Naive Bayes
HIDS Host Intrusion Detection Systems
IA Intelligence Artificielle
IdO Internet des objets
IDS Intrusion detection System
IoT Internet of Things
KNN k-Nearest Neighbours
LDA Latent Dirichlet Allocation
LR Logistic Regression
LSTM Long Short-Term Memory
MITM Man-in-the-Middle
ML Machine Learning
MLP Multi layer perceptron
NaN Not a Number
NIDS Network Intrusion Detection Systems
OSI Open Systems Interconnection
OSVM One-Class Support Vector Machine
PCA Principal Component Analysis
RFC Random Forest Classifier
RNN Recurrent Neural Networks
SMOTE Synthetic Minority Over-sampling Technique
SVM Support Vector Machine
TN True Negative
TP True Positive
t-SNE t-distributed Stochastic Neighbor Embedding
WSN Wireless Sensors Network

Introduction générale

L'avènement des outils numériques et leur intégration croissante à tous les aspects de notre vie quotidienne ont ouvert de nouvelles perspectives et ont profondément transformé notre société connectée. L'internet des objets (IoT) et le machine learning (ML) sont des technologies clés qui ont joué un rôle majeur dans cette révolution numérique. Cependant, cette évolution rapide n'est pas sans poser de nouveaux défis en matière de sécurité informatique. Les menaces qui pèsent sur les systèmes informatiques ont également évolué, nécessitant des solutions innovantes pour garantir la protection des données personnelles et la sécurité des systèmes.

Le présent mémoire se concentre sur l'utilisation du machine learning et du deep learning dans le domaine de la sécurité, en particulier dans le contexte de l'IoT. L'IoT offre la possibilité de connecter des objets de notre vie quotidienne à internet, créant ainsi un environnement où les objets peuvent interagir entre eux et avec les utilisateurs [1]. Cependant, cette interconnexion constante expose ces objets à de nouvelles vulnérabilités et à des risques de sécurité accrus.

L'une des principales menaces auxquelles fait face l'IoT est représentée par les attaques botnets, qui peuvent être utilisées pour mener diverses attaques malveillantes, telles que les attaques par déni de service distribué (DDoS) ou les attaques de type ransomware. Les systèmes de détection d'intrusion (IDS) sont des outils essentiels pour identifier ces activités malveillantes et protéger les systèmes IoT. Cependant, les techniques traditionnelles de détection d'intrusion peuvent se révéler insuffisantes face à la sophistication croissante des attaques.

C'est là qu'intervient le machine learning. En permettant aux systèmes de détection d'intrusion d'apprendre à partir des données de trafic réseau, le machine learning offre une approche prometteuse pour détecter et prévenir les attaques botnets. Les algorithmes de machine learning peuvent analyser de vastes quantités de données, détecter des schémas anormaux et identifier les comportements malveillants, même s'ils n'ont jamais été rencontrés auparavant [2]. De plus, le deep learning, une branche du machine learning basée sur les réseaux de neurones profonds, offre des capacités encore plus avancées pour la détection des menaces.

Dans ce mémoire, nous proposons une approche spécifique pour la détection des attaques botnets dans les systèmes IoT en utilisant des techniques de machine learning et de deep learning. Nous explorons différentes méthodes et algorithmes, et nous évaluons leur efficacité et leurs performances à travers des expérimentations approfondies. Notre objectif est de contribuer à renforcer la sécurité des systèmes IoT en fournissant une solution innovante et efficace pour détecter et prévenir les attaques botnets.

La suite de ce mémoire est organisée de la manière suivante :

Le premier chapitre est divisé en trois parties. La première partie aborde la sécurité informatique, les attaques et les menaces. La deuxième partie présente le machine learning et le deep learning, ainsi que leurs différentes techniques. Enfin, la dernière partie est consacrée à

l'internet des objets (IoT).

Dans le second chapitre, nous présentons une revue de la littérature sur les attaques botnets, les techniques de détection existantes et les travaux de recherche connexes.
Enfin dans le dernier chapitre, nous détaillons notre approche proposée, en expliquant les différentes étapes et les algorithmes utilisés. Nous présentons ensuite les résultats de nos expérimentations et analysons les performances de notre approche.

Finalement, nous concluons en soulignant les contributions apportées par notre travail, en discutant de ses limites et en proposant des pistes pour des recherches futures dans ce domaine crucial de la sécurité des systèmes IoT.

Chapitre 1

Définitions et concepts de base

Introduction

Le machine learning est un domaine d'intelligence artificielle qui permet aux ordinateurs d'apprendre à partir de données, sans être explicitement programmés. Cette technologie peut être utilisée pour améliorer la sécurité informatique en détectant et prévenant les cyberattaques, en identifiant les vulnérabilités et en renforçant les mécanismes de sécurité. La puissance du Machine Learning dans la sécurité prend une autre dimension dans l'IoT, cette technologie en essor depuis plus d'une décennie étant capable de collecter et d'analyser des quantités massives de données provenant de divers objets connectés. Cette synergie entre le Machine Learning et l'IoT offre ainsi de grandes opportunités pour améliorer la sécurité dans de nombreux secteurs tels que la santé, la maison connectée, l'industrie ou encore les villes intelligentes, et qui regroupe des objets connectés à internet, tels que des capteurs, des caméras de surveillance, des thermostats, etc. Entre autre, le machine learning permet d'améliorer la sécurité en détectant les comportements malveillants ou les anomalies, en analysant les données en temps réel et en prenant des mesures préventives pour réduire les risques de sécurité. Cependant, le machine learning peut également présenter des défis en matière de sécurité pour l'IoT, car les objets connectés peuvent être vulnérables aux attaques et aux piratages, et les modèles de machine learning peuvent être susceptibles aux attaques visant à manipuler les données ou à les rendre imprécises. Le machine learning peut être utilisé pour améliorer la sécurité informatique et de l'IoT, mais il est également important de comprendre les risques potentiels associés à cette technologie et de mettre en place des mesures de sécurité appropriées pour protéger les systèmes.
Ce chapitre est organisé en trois sections : La section 1.1 décrit un aperçu sur la sécurité informatique. La section 1.2 définit le machine learning et sa relation à la sécurité. La dernière section 1.3 définit des notions de base sur l'Internet des objets.

1.1 Sécurité informatique

Dans cette section, nous aborderons la sécurité informatique et ses objectifs. Nous exposerons l'évolution des services de sécurité. Nous discuterons également des menaces courantes auxquelles les entreprises et les particuliers peuvent être confrontés, ainsi que des procédures de sécurité qui peuvent être mises en place pour minimiser les risques.

1.1.1 Définition de la sécurité informatique

La sécurité informatique est un domaine d'expertise utilisant un ensemble de techniques et de méthodes visant à prévenir, à détecter et à protéger les systèmes informatiques, les réseaux, les applications, les données et les utilisateurs contre les menaces qui peuvent les compromettre.Ça peut être un virus, des logiciels malveillants, des attaques par déni de service, des violations de données et des cyberattaques [3].
Les mesures de sécurité prisent en compte peuvent inclure l'utilisation de pares-feux, de systèmes de détection d'intrusion, de cryptage des données, de politiques de sécurité, de procédures de gestion des incidents, de formation des utilisateurs, de tests de pénétration et d'audits de sécurité.

1.1.2 Objectifs de la sécurité dans l'ido

Le principal but de la sécurité informatique est la protection contre les menaces et ceci en garantissant :

- **La confidentialité :** Protéger les données collectées, transmises et stockées par les objets connectés contre tout accès non autorisé [4].
- **L'intégrité :** L'objectif de l'intégrité est de s'assurer que les données et les informations ne sont pas modifiées de manière non autorisée. Cela peut être réalisé grâce à des mécanismes de contrôle d'accès, de journalisation et de sauvegarde [4].
- **Sécurité du réseau :** Mettre en place des mesures de sécurité pour protéger les communications entre les objets connectés, les passerelles IoT, les plateformes de gestion et les utilisateurs finaux.
- **L'authentification :** Mettre en place des mécanismes d'identification robustes pour s'assurer que seuls les utilisateurs autorisés peuvent accéder aux objets connectés et aux données qui y sont associées.
- **Gestion des vulnérabilités :** Identifier, évaluer et corriger les vulnérabilités de sécurité dans les objets connectés, les protocoles de communication et les systèmes IoT pour réduire les risques d'exploitation [4].
- **Détection des intrusions :** Mettre en place des systèmes de surveillance et de détection des activités malveillantes ou suspectes dans l'écosystème IoT pour identifier les tentatives d'intrusion et y répondre rapidement.

1.1.3 Évolution des services de sécurité informatique

Les services de sécurité informatique ont évolué au fil du temps en réponse aux nouvelles menaces et aux besoins des organisations. Voici quelques exemples d'évolution des services de sécurité informatique ;

1. **Passage de la prévention à la détection et à la réponse :** Les services de sécurité informatique ont évolué d'une approche axée principalement sur la prévention des attaques

à une approche plus pro-active axée sur la détection précoce des intrusions et la réponse rapide aux incidents de sécurité.

2. **Utilisation de l'analyse comportementale :** les services de sécurité informatique utilisent de plus en plus l'analyse comportementale pour détecter les menaces. Cela implique de surveiller les comportements d'utilisation normaux et d'alerter les utilisateurs ou les administrateurs si des comportements suspects sont détectés.
3. **Utilisation de l'intelligence artificielle et de l'apprentissage automatique :** l'utilisation de l'intelligence artificielle et de l'apprentissage automatique est de plus en plus courante dans les services de sécurité informatique. Ces technologies peuvent aider à détecter les menaces plus rapidement et plus efficacement en apprenant de nouveaux comportements suspects.
4. **Prise en compte de la conformité réglementaire :** Avec l'introduction de réglementations telles que le RGPD (Règlement général sur la protection des données), les services de sécurité informatique doivent également se conformer à des exigences légales en matière de protection des données et de confidentialité.
5. **Utilisation de la sécurité basée sur le cloud :** les services de sécurité informatique sont de plus en plus fournis sous forme de services basés sur le cloud. Cela permet aux organisations de bénéficier de la sécurité sans avoir à gérer les infrastructures de sécurité elles-mêmes.

1.1.4 Types de menaces sur les systèmes informatique

La menace relève de tout ce qui peut potentiellement nuire aux systèmes informatiques. On a deux types de menaces de sécurité réseau :

— **Menaces actives :** c'est une tentative d'accès non autorisé à l'ordinateur par l'attaquant. Elles impliquent des modifications ou des changements durant une communication [5].
— **Menaces passives :** impliquent une tentative d'intrusion à un système ou un accès non-autorisé aux données de l'utilisateur sans pour autant modifier la communication [5].

Cependant il ne faut pas confondre menaces, failles et attaques. Une faille informatique est une vulnérabilité dans un système qui peut être exploitée par une personne malveillante pour accéder à des informations sensibles ou pour perturber le fonctionnement normal du système. Par exemple, une faille de sécurité dans un logiciel de traitement de texte pourrait permettre à un attaquant d'exécuter un code malveillant sur un ordinateur,tandis qu'une menace informatique est une situation qui peut compromettre la sécurité d'un système,et l'attaque informatique est une action malveillante visant à compromettre la sécurité d'un système.

Il existe de nombreuses menaces et attaques qui peuvent affecter les réseaux. Certaines sont mentionnées ci-dessous :

1. **Virus informatiques :** les virus informatiques sont des programmes ou des fragments de programmes malveillants qui infectent les systèmes informatiques en se propageant à travers les fichiers et les programmes.
2. **Logiciels malveillants :** les logiciels malveillants sont des programmes conçus pour endommager les systèmes informatiques ou voler des données. Les exemples incluent les chevaux de Troie, les vers et les ransomwares.
3. **Spyware :** Un spyware ou logiciel espion est un type de logiciel malveillant conçu pour collecter des informations sur un ordinateur ou un utilisateur sans son consentement.

Les spywares sont souvent installés à l'insu de l'utilisateur lorsqu'il télécharge ou installe un programme tiers. Une fois installé, le spyware peut surveiller les habitudes de navigation sur Internet, collecter des informations personnelles, comme les mots de passe, les numéros de carte de crédit et d'autres informations sensibles [6].

4. **Attaques par déni de service (DoS) :** les Des attaquants tentent de submerger un objet connecté, un réseau ou une infrastructure IoT avec un trafic excessif, ce qui peut entraîner une interruption de service pour les utilisateurs légitimes. La figure ci-dessous démontre un scénario d'attaque DoS :

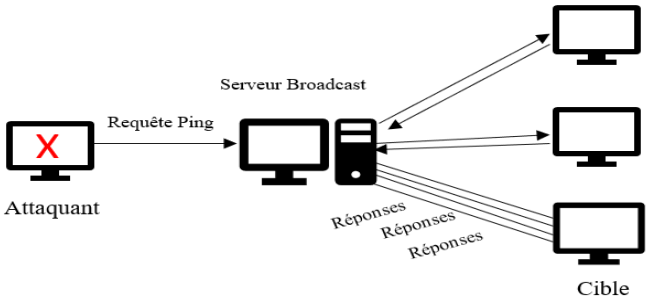

FIGURE 1.1 – Illustration d'une attaque par déni de service (DoS)
[7]

5. **Attaques d'ingénierie sociale :** les attaques d'ingénierie sociale sont des attaques qui visent à tromper les utilisateurs pour qu'ils divulguent des informations sensibles ou qu'ils installent des logiciels malveillants en utilisant des techniques de manipulation psychologique.
6. **Attaque de l'homme au milieu (Man-in-the-middle) :** est une attaque informatique où un attaquant intercepte la communication entre deux parties pour lire, modifier ou injecter des données malveillantes. L'attaquant peut agir de manière passive ou active et cela peut conduire au vol d'informations sensibles comme des identifiants de connexion, des données bancaires ou des informations personnelles. La figure suivante illustre le scénario d'une attaque de l'homme au milieu (MITM) :

FIGURE 1.2 – Illustration d'une attaque de MITM
[7]

Ces menaces peuvent avoir de graves conséquences pour les entreprises et les particuliers, allant de la perte de données et des atteintes à la vie privée à la perte de revenus et à l'atteinte à la réputation. Par conséquent, il est important de mettre en œuvre des mesures de sécurité pour protéger les systèmes informatiques de ces menaces.

1.1.5 Attaques de Botnet dans l'iot

Les attaques de botnets dans l'Internet des objets (IoT) sont des attaques où un ensemble d'objets connectés compromis, appelés "bots", sont utilisés pour mener des actions malveillantes coordonnées [8]. Les botnets IoT sont formés en infectant un grand nombre d'objets connectés avec des logiciels malveillants spécialement conçus.

Une fois qu'un botnet IoT est établi, les attaquants peuvent exercer un contrôle sur les objets connectés infectés et les utiliser pour mener divers types d'attaques, tels que :

- **Attaque DDoS (distributed denial-of-service) :** Les bots du botnet inondent une cible spécifique, telle qu'un site web ou une infrastructure réseau, avec un trafic excessif, le submergeant et le rendant indisponible pour les utilisateurs légitimes.
- **Spam et phishing :** Les bots du botnet peuvent être utilisés pour envoyer un grand nombre de courriers indésirables (spam) ou des messages de phishing, visant à tromper les destinataires pour obtenir leurs informations personnelles ou financières.
- **Fraude publicitaire :** Les bots peuvent cliquer automatiquement sur des publicités en ligne, générant ainsi des revenus publicitaires frauduleux pour les attaquants.
- **Attaques de force brute :** Les bots du botnet peuvent tenter de deviner ou de craquer des mots de passe en essayant différentes combinaisons, ce qui peut compromettre des comptes ou des systèmes.
- **Mining de cryptomonnaie :** Les bots peuvent être utilisés pour extraire illégalement des cryptomonnaies, utilisant les ressources informatiques des objets connectés infectés sans le consentement des propriétaires.
- **Espionnage et surveillance :** Les bots peuvent collecter des informations sensibles à partir des objets connectés infectés, tels que des données personnelles, des enregistrements audio ou des flux vidéo.

L'une des principales caractéristiques des attaques de botnets IoT est l'échelle massive des objets connectés infectés, ce qui peut entraîner des conséquences dévastatrices en termes de perturbation des services en ligne, de vol d'informations sensibles et de compromission de la vie privée. Pour se protéger contre les attaques de botnets IoT, il est essentiel de mettre en œuvre des mesures de sécurité solides.

1.1.6 Mécanismes de réponses aux menaces et attaques informatique

Les mécanismes de réponse aux menaces et attaques informatiques sont un ensemble de procédures et d'outils qui permettent de détecter, de gérer et de résoudre les incidents de sécurité. On peut distinguer deux approches :

- **Approche réactive :** consiste à réagir aux problèmes de sécurité une fois qu'ils se sont produits. Cela peut inclure :
  1. **Anti-virus et anti-malware :** ces logiciels sont utilisés pour détecter et éliminer les virus et les logiciels malveillants.
  2. **IDS :** Un système de détection d'intrusion est un système de sécurité informatique qui surveille les activités du réseau ou d'un système informatique pour détecter toute activité malveillante ou non autorisée [9]. Les IDS peuvent être classés selon deux types :
     - **IDS basé sur l'emplacement :**
       - IDS basé sur l'hôte (HIDS) : est un système de détection d'intrusion, qui surveille et analyse les activités et les fichiers système d'un ordinateur ou d'un

objet connecté pour détecter les tentatives d'intrusion et les comportements malveillants. L'emplacement de ce type d'ids est illusté dans la figure suivante :

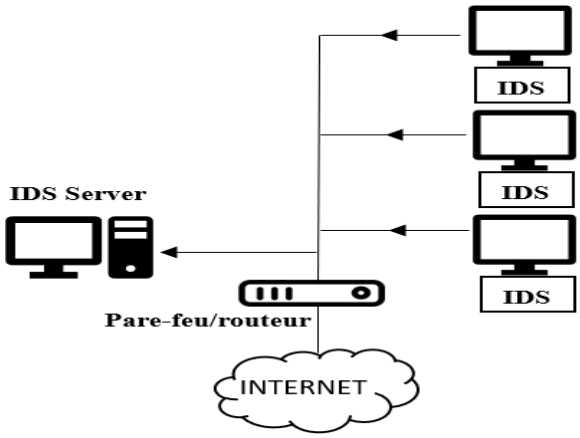

FIGURE 1.3 – Illustration d'un HIDS

- IDS basé sur le réseau (NIDS) : est un système de détection d'intrusion, qui analyse le trafic réseau pour identifier les activités suspectes, les attaques et les comportements malveillants.L'emplacement de ce type d'ids est illusté dans la figure suivante :

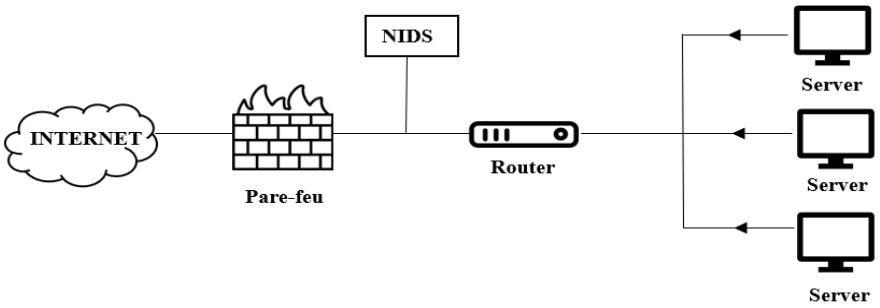

FIGURE 1.4 – Illustration d'un NIDS

- IDS Hybride :est un système de détection d'intrusion qui combine à la fois des fonctionnalités de détection basées sur le réseau (NIDS) et des fonctionnalités de détection basées sur l'hôte (HIDS) pour une meilleure couverture de détection des menaces et des attaques.

— **IDS basé sur la méthode de détection :**
- IDS basés sur les signatures : ces IDS se basent sur des signatures préalablement identifiées de comportements malveillants, pour identifier les menaces similaires dans les données entrantes.
- IDS basés sur le comportement : ces IDS analysent les activités d'un système ou d'un utilisateur pour identifier des comportements anormaux. Ils peuvent également utiliser l'apprentissage automatique pour détecter des motifs de comportement malveillants.

— **Sécurité des réseaux sans fil :** la sécurité des réseaux sans fil comprend des mécanismes tels que l'authentification des utilisateurs et la segmentation des réseaux.

— **Surveillance des journaux d'activité :** la surveillance des journaux d'activité des systèmes et des applications peut aider à détecter les activités suspectes et les tentatives d'attaque. En surveillant les journaux d'activité, il est possible de détecter rapidement les incidents de sécurité et d'agir en conséquence.

- **Approche préventive :** consiste à mettre en place des mesures pro-actives pour réduire les risques de sécurité avant qu'ils ne surviennent. Cela inclut des mesures tels que :

1. **Pare-feu :** un pare-feu est une barrière de sécurité qui bloque le trafic non autorisé entre un réseau privé et Internet. La figure ci-dessous illustre un pare-feu :

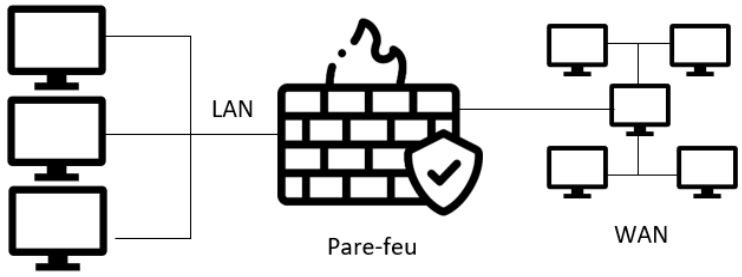

FIGURE 1.5 – Illustration d'un Pare-feu

2. **Cryptographie :**la cryptographie permet de chiffrer les données afin qu'elles ne soient pas accessibles à des tiers non autorisés.
3. **Contrôle d'accès :** le contrôle d'accès permet de définir les autorisations d'accès à un système ou à des données pour les utilisateurs et les groupes.
4. **Sécurité physique :** la sécurité physique comprend des mesures telles que la surveillance vidéo, les portes à serrure électronique et les barrières pour empêcher l'accès non autorisé aux locaux.
5. **Gestion des identités et des accès :** la gestion des identités et des accès (IAM) permet de gérer les identités des utilisateurs et leur accès aux systèmes et aux données.
6. **Tests de pénétration :** les tests de pénétration sont des tests effectués pour évaluer la sécurité d'un système en simulant une attaque.

1.2 Apprentissage automatique

Dans cette section nous allons aborder quelques éléments préliminaire du domaine de l'apprentissage automatiques qui seront cruciaux pour la compréhension de notre travail.

1.2.1 Définition de l'apprentissage automatique

L'apprentissage automatique ou machine learning est une branche évolutive de l'intelligence artificielle qui donne aux ordinateurs la possibilité d'apprendre sans être explicitement programmé. Les algorithmes capables d'apprendre à partir de données et de prendre des décisions sur la base de modèles observés dans ces données [10]. En d'autres termes, on dit qu'un programme informatique apprend de l'expérience E en ce qui concerne une tache T et une mesure de performance P si sa performance sur T mesuré par P s'améliore avec l'expérience E.

1.2.2 Types des systèmes de l'apprentissage automatique

Il existe plusieurs types de systèmes de machine learning qu'on peut les classer selon différentes catégories majeures :

- Selon l'effectuation de l'apprentissage progressivement, au fur et à mesure ou non (Apprentissage en ligne,Apprentissage groupé).
- Selon s'il compare uniquement les nouvelles données à des données connues ou qu'il détecte les éléments de structuration dans les données d'entraînement et construit un modèle prédictif comme un scientifique (Apprentissage à partir d'observation,Apprentissage à partir d'un modèle).
- Selon l'effectuation de l'apprentissage avec ou sans supervision humaine (Apprentissage non-supervisé, Apprentissage semi-supervisé, Apprentissage supervisé, Apprentissage par renforcement).
  - **Apprentissage non-supervisé :** Dans l'apprentissage non supervisé, l'environnement ne fournit que des données d'entrée sans objectifs souhaités. Il n'a pas besoin de données étiquetées, le système tente d'apprendre sans professeur et peut étudier les similitudes entre les données non étiquetées et classer les données en différents groupes. Les méthodes classent les objets sans besoin d'une base de données [11]. On distingue plusieurs types à savoir le clustering qui permet de regrouper les données similaires, la réduction de dimension,etc.
  - **Apprentissage semi-supervisé :** l'apprentissage semi-supervisé se situe entre l'apprentissage supervisé et celui non supervisé, il utilise à la fois des données labellisées et non labellisées pour s'adapter à un modèle.
  - **Apprentissage supervisé :** L'apprentissage automatique peut être supervisé dans le cas où des étiquettes sont associées à des données d'apprentissage, et vous essayez de prédire des étiquettes pour des données futures. Autrement dit les méthodes consistent à classer les objets à partir d'une base de données dite d'apprentissage.
    Avec le Supervised Learning on peut développer des modèles pour résoudre deux types de problèmes : les problèmes de *Régression* ou les problèmes de *Classification* [11].
    Dans le premier cas on cherche à faire une prédiction d'une variable continue, par exemple prédire le prix d'une maison (y) selon une surface habitable (x). On appelle ces variables des prédicteurs, et les valeurs qu'on veut prédire sont appelées les caractéristiques comme illustré ci-dessous :

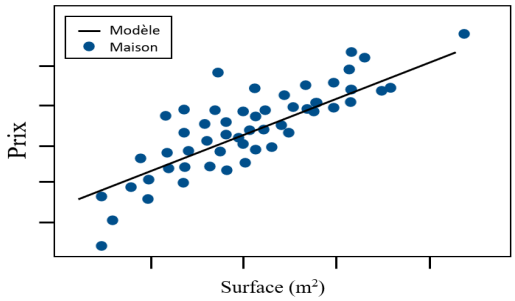

FIGURE 1.6 – Régression

Dans le second cas on cherche à classer un objet dans différentes classes, c'est-à-dire faire la prédiction d'une variable discrète (prend un nombre fini de valeurs), comme exemple prédire si un email est un spam (classe y= 1) ou non (classe y = 0) selon le nombre de liens présent dans l'email (x) ; le filtre anti-spam est formé avec de nombreux exemples d'emails avec leur classe : spam (1) ou Ham (0), et il

doit apprendre à classer les nouveaux emails. Dans le cas de supervisé on donne à l'ordinateur les données et le résultat et lui va fournir le modèle, comme démontré dans la figure suivante :

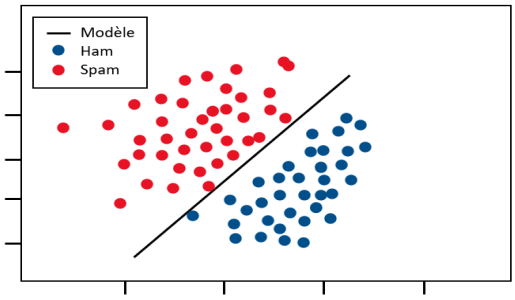

FIGURE 1.7 – Classification

— **Apprentissage par renforcement :** Contrairement au scénario d'apprentissage supervisé, le scénario de renforcement implique des essais et des erreurs pour maximiser la récompense de l'agent. Ainsi, le but de ce dernier est de déterminer le meilleur plan d'action ou la meilleure stratégie pour atteindre l'objectif. Un apprenant ne reçoit pas passivement un ensemble de données étiquetées. Au lieu de cela, il recueille des informations à travers une série d'actions qui interagissent avec l'environnement.

La figure ci-dessous résume les différents types de système de l'apprentissage automatique qui ont été mentionné ci-dessus, selon la manière dont l'apprentissage a été effectué ;

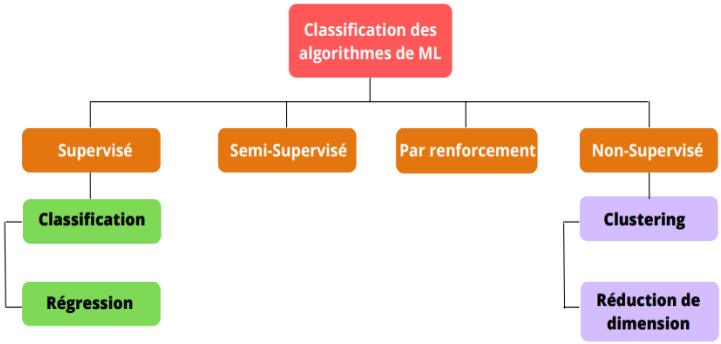

FIGURE 1.8 – Classification selon l'effectuation de l'apprentissage (avec ou sans supervision humaine)

1.2.3 Techniques de l'apprentissage automatique

Il existe plusieurs algorithmes de machine learning parmi eux nous en citons les suivants :

- **Algorithmes supervisé :** Les algorithmes d'apprentissage supervisé les plus courants sont les suivants :
  — **Arbre de décision :**l'arbre de décision est l'un des algorithmes d'apprentissage supervisé qui est utilisé pour effectuer des taches de la classification et aussi pour la régression, Les arbres de décision sont également les composants fondamentaux des forêts aléatoires, qui comptent parmi les algorithmes d'apprentissage automatique les plus puissants disponibles aujourd'hui. Le modèle est composé des nœuds, feuilles et

branches ; chaque nœud représente une caractéristique ou un attribut. Les branches représentent les règles, tandis que la feuille représente une étiquette de classe (un résultat possible). Voici un exemple pour prédire si une personne aura besoin ou non d'un traitement à travers quelques questions, comme illustré dans la figure suivante :

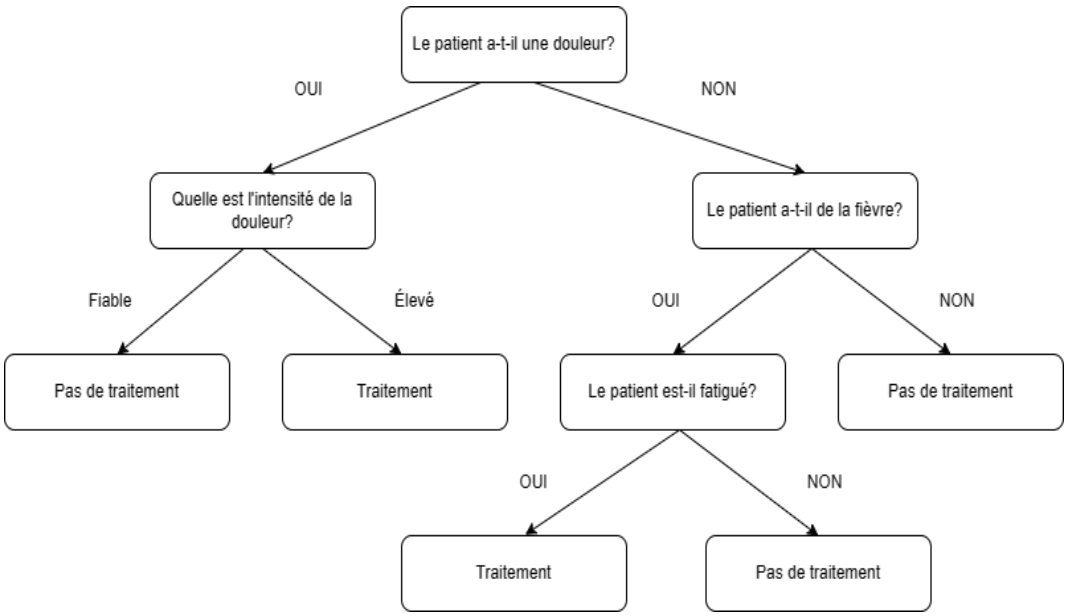

FIGURE 1.9 – Classification de la nécessite de prescrire un traitement ou non pour une personne avec l'arbre de décision

— **Forêts aléatoires :** C'est une collection d'un grand nombre d'arbre de décision individuels mais fonctionnant comme un ensemble ou chaque arbre fait une prédiction de classe, et la classe qui a plus de votes va être considérer comme prédiction finale de notre modèle.

— **K-plus proches voisins (KNN) :** Est l'un des algorithmes supervisés aussi, qui identifie un échantillon en fonction de sa distance par rapport à ses voisins. Nous pouvons prendre un exemple de classification d'une plante si elle est une Sétosa ou une versicolore en prenant compte la largeur et la langueur de sa feuille et aussi ses voisins, si on prend les 7 plus proches voisins et on trouve que 5 parmi eux sont des sétosa et 2 des versicolore alors on dit que la plante est une sétosa, comme figure ci-dessous :

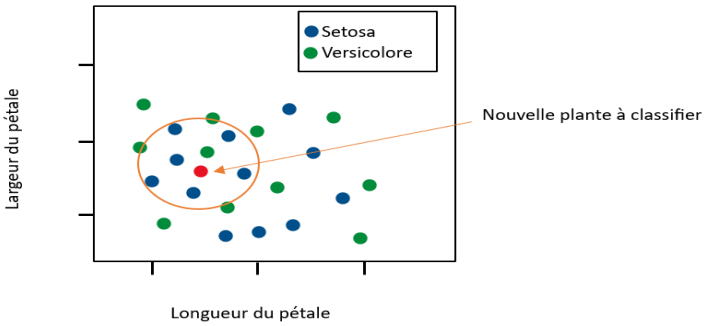

FIGURE 1.10 – Classification de plante avec KNN

— **Régression linéaire :** est une technique d'apprentissage supervisé utilisée pour modéliser la relation entre une variable dépendante continue et une ou plusieurs variables

indépendantes continues ou catégorielles. L'objectif de la régression linéaire est de trouver une fonction linéaire qui minimise l'erreur entre les prédictions du modèle et les valeurs réelles de la variable dépendante [12].

— **Régression logistique :** est une technique d'apprentissage supervisé utilisée pour résoudre des problèmes de classification binaire ou multiclasse. Elle est similaire à la régression linéaire, mais la variable dépendante est une variable binaire (0 ou 1) ou une variable catégorielle avec plusieurs niveaux. Le modèle de régression logistique utilise une fonction logistique pour calculer la probabilité qu'une observation appartienne à une classe particulière en fonction des valeurs de ses caractéristiques [13]. La fonction logistique transforme les valeurs de l'entrée en une probabilité, qui est comprise entre 0 et 1.

— **Naive Bayes :** L'algorithme Naive Bayes est un algorithme d'apprentissage automatique supervisé basé sur le théorème de Bayes. Il est principalement utilisé pour la classification de textes et d'autres types de données discrètes. Naive Bayes suppose une indépendance conditionnelle entre les caractéristiques, ce qui signifie qu'il considère que chaque caractéristique contribue indépendamment à la probabilité d'appartenance à une classe. Il est souvent utilisé pour la détection de spam, la classification de documents et la catégorisation de texte.

— **Machines à vecteurs de support (SVM) :** Une machine à vecteur de support (SVM) est un algorithme d'apprentissage automatique supervisé utilisé généralement pour la classification basée sur l'idée d'un hyperplan de séparation dans un espace de caractéristiques. Ci-dessous un exemple de classification binaire en utilisant SVM :

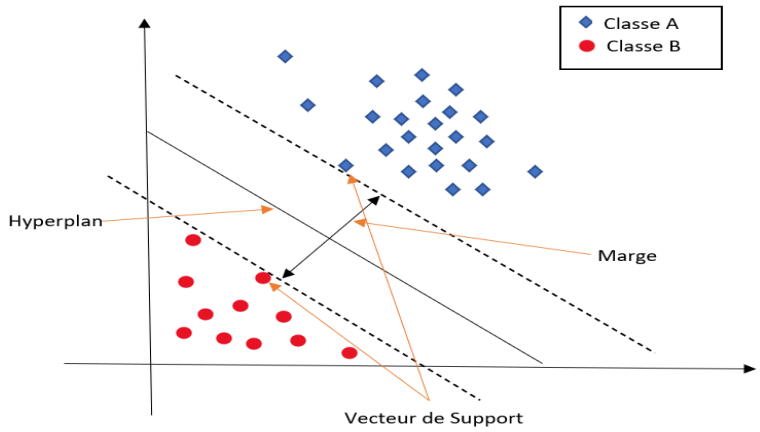

FIGURE 1.11 – Illustration d'un exemple avec SVM [14]

- **Algorithmes non-supervisé :** Les algorithmes d'apprentissage non-supervisé les plus courants sont les suivants :

— **Partitionnement hiérarchique :** est une technique de clustering (ou regroupement) de données qui consiste à diviser successivement un ensemble de données en groupes plus petits et plus homogènes. Il existe deux types de partitionnement hiérarchique : le partitionnement hiérarchique ascendant et le partitionnement hiérarchique descendant [15].

— **LDA (Latent Dirichlet Allocation) :** L'algorithme LDA est une méthode d'apprentissage automatique non supervisée utilisée pour la modélisation de sujets. Il est utilisé pour découvrir des thèmes ou des sujets cachés à partir d'un ensemble de documents. LDA suppose que chaque document est une combinaison de différents sujets et que chaque mot dans un document est attribué à un certain sujet [16].

- **DBSCAN :** L'algorithme DBSCAN (Density-Based Spatial Clustering of Applications with Noise) est une technique de clustering de données qui permet d'identifier des groupes de points de données dans un espace de dimensions élevées [17]. L'algorithme DBSCAN est particulièrement efficace pour trouver des clusters de formes arbitraires et pour détecter les points aberrants.
- **PCA (Principal Component Analysis) :** l'Analyse en Composantes Principales permet de transformer un ensemble de données multidimensionnelles en un ensemble de données à moins de dimensions, tout en préservant les informations les plus significatives. Cette réduction de dimensionnalité est obtenue en projetant les données sur un nouvel ensemble de variables appelées les composantes principales [18]. Les composantes principales sont calculées de manière à ce que chaque composante soit orthogonale (non corrélée) et ordonnée en fonction de l'importance de leur contribution à la variance totale des données.
- **Méthode t-SNE (t-distributed Stochastic Neighbor Embedding) :** est une technique de réduction de dimensionnalité non linéaire utilisée pour représenter des données de haute dimension dans un espace de dimension plus faible tout en préservant les relations locales entre les points de données [18]. Cette technique est particulièrement utile pour visualiser des données complexes dans un espace de deux ou trois dimensions.
- **K-means (K-moyen) :** La méthode K-moyen est l'un des algorithmes d'apprentissage non supervisé, son fonctionnement est comme suit : Après avoir fixé le nombre K de cluster l'algorithme va choisir les k centres et les objets, ensuite on calcule la distance entre les objets et chaque cluster et on les affectes au cluster le plus proche, les centres vont être redéfinis à nouveau, l'algorithme se répète donc jusqu'à ce qu'il y ait convergence (d'une autre manière jusqu'à ce que l'on trouve que les centres des clusters ne changent pas). Voici un exemple illustratif avec l'algorithme K-means :

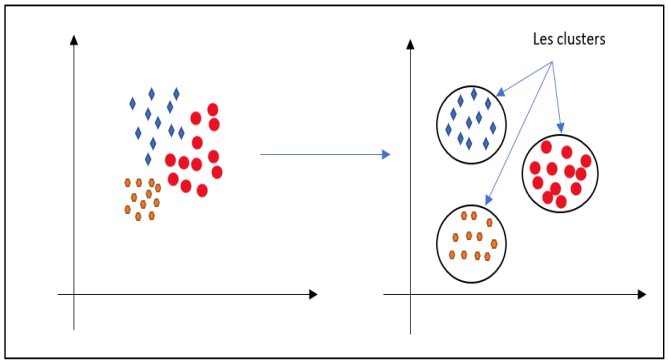

FIGURE 1.12 – Exemple illustré avec K-means

- **Méthodes d'ensemble :** consiste en la combinaisons de plusieurs modèles pour améliorer la précision et la robustesse des prédiction. Voici quelques-unes des plus courantes :
  - **Bagging :** cette méthode consiste à entraîner plusieurs modèles sur des échantillons aléatoires avec remplacement des données d'entraînement. Puis à combiner les prédictions de chaque modèle pour plus de robustesse.
  - **Boosting :** cette méthode entraîne une série de modèles faibles en accentuant les erreurs de prédiction du modèle précédent.
  - **Stacking :** cette méthode consiste à entraîner plusieurs modèles de différents types sur les mêmes données d'entraînement. Les prédictions de chaque modèle sont ensuite utilisées comme entrée pour un autre modèle qui prédit la sortie finale.

— **Vote majoritaire :** dans cette méthode le résultat final est obtenu suite à un vote par plusieurs modèles sur les prédictions, on retient seulement celle voté à la majorité.

1.2.4 Application de l'apprentissage automatique à la sécurité

le machine learning peut être utilisé pour améliorer la sécurité de divers systèmes et applications, et est souvent utilisé en conjonction avec d'autres techniques de sécurité pour fournir une protection plus complète contre les menaces.Il peut être appliqué à la sécurité de différentes manières, tel que :

— **Détection d'anomalies :** Le machine learning peut être utilisé pour détecter des comportements ou des événements anormaux dans les systèmes de sécurité informatique. En apprenant à partir de données historiques, les algorithmes de machine learning peuvent identifier des modèles d'activité qui ne sont pas conformes à la normale, tels que des connexions suspectes à un réseau ou des tentatives d'accès à des ressources protégées.
— **Prévention des fraudes :** Le machine learning peut également être utilisé pour prévenir les fraudes en ligne, telles que les fraudes par carte de crédit ou les attaques de phishing. En utilisant des techniques d'apprentissage supervisé, les algorithmes peuvent apprendre à identifier les schémas et les caractéristiques des transactions frauduleuses et les bloquer avant qu'elles ne se produisent.
— **Détection des menaces :** En utilisant des techniques d'apprentissage supervisé et non supervisé, les algorithmes de machine learning peuvent identifier les menaces potentielles, tels que les logiciels malveillants, les virus et les attaques de déni de service.
— **Sécurité des réseaux :** Le machine learning peut également être utilisé pour améliorer la sécurité des réseaux, par exemple en détectant les anomalies dans les flux de données réseau ou en identifiant les comportements suspects des utilisateurs.

1.2.5 Apprentissage profond

L'apprentissage profond ou le deep learning est une technique de l'apprentissage automatique qui utilise des réseaux de neurones artificiels pour apprendre à partir de données. Les réseaux de neurones sont des modèles mathématiques qui simulent le fonctionnement du cerveau humain en apprenant à partir de données brutes et en effectuant des tâches de classification, de reconnaissance d'images, de traitement du langage naturel et bien plus encore [19]. Le deep learning est également utilisé dans de nombreux domaines de recherche, notamment en biologie, en physique, en astronomie et en économie, où il est utilisé pour extraire des informations à partir de données massives.

1.2.5.1 Structure d'un neurone artificiel

Un neurone artificiel est une unité de base du réseau de neurones artificiels. Il est conçu pour imiter le fonctionnement des neurones biologiques du cerveau. La structure d'un neurone artificiel est composée de trois parties principales :

— **Les entrées :** Les entrées sont les signaux qui sont envoyés au neurone. Chaque entrée est associée à un poids qui détermine l'importance de cette entrée pour le neurone.
— **La fonction d'activation :** La fonction d'activation est une fonction mathématique qui transforme la somme pondérée des entrées en une sortie. Il existe plusieurs types de fonctions d'activation, telles que la fonction sigmoïde, la fonction tanh ou la fonction ReLU.
— **La sortie :** La sortie est le résultat produit par le neurone, après avoir appliqué la fonction d'activation sur la somme pondérée des entrées, comme illustré ci-dessous :

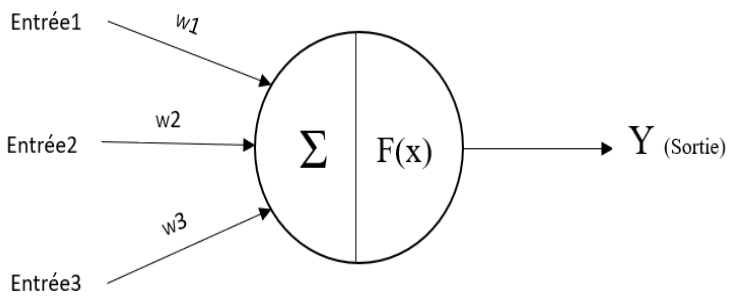

FIGURE 1.13 – Structure d'un neurone artificiel
[20]

Le processus de calcul d'un neurone artificiel est le suivant : les entrées sont multipliées par leur poids associé, puis ces résultats sont sommés pour produire une valeur numérique. Cette valeur est ensuite entrée dans la fonction d'activation pour produire la sortie.

1.2.5.2 Modèles des réseaux de neurones artificiels

Nous allons présenter dans cette section les modèles de réseaux de neurones artificiels les plus courants :

- **Perceptron :** Il s'agit du modèle de base d'un neurone artificiel, qui prend des entrées pondérées, applique une fonction d'activation et produit une sortie.
- **Multi-Layer Perceptron (MLP) :** C'est un réseau de neurones artificiels avec plusieurs couches de neurones, y compris une couche d'entrée, une ou plusieurs couches cachées et une couche de sortie. Chaque neurone est connecté à tous les neurones de la couche suivante [21], la figure suivante illustre la structure d'un perceptron multi-couches :

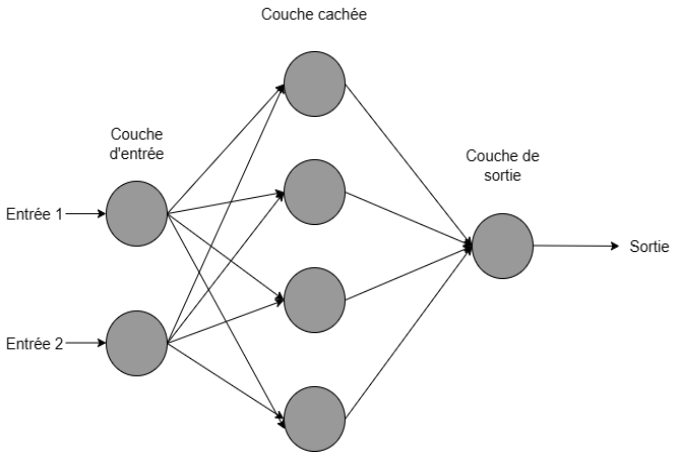

FIGURE 1.14 – Structure d'un MLP
[20]

- **Réseaux de neurones convolutifs (Convolutional Neural Networks, CNN) :** Les réseaux de neurones convolutifs sont un type de réseau de neurones utilisé pour l'analyse d'images, de vidéos et de sons. Les CNN sont conçus pour reconnaître des motifs et des caractéristiques dans des données visuelles, en utilisant des couches de convolution qui filtrent et extraient des informations de l'image en entrée [21].
- **Réseaux de neurones récurrents (Recurrent Neural Networks, RNN) :** Les réseaux de neurones récurrents sont un type de réseau de neurones qui peuvent prendre en compte l'historique de données précédentes pour générer des prédictions ou des classifications.

Les RNN sont souvent utilisés pour l'analyse de données séquentielles, comme du texte ou des séquences temporelles [21]. Cependant, les RNN souffrent souvent de problèmes de rétro-propagation du gradient qui peuvent limiter leur capacité à traiter des séquences de données longues.

— **Réseaux de neurones à mémoire à court terme (Long Short-Term Memory, LSTM) :** Les réseaux de neurones à mémoire à court terme (LSTM) sont une variante des réseaux de neurones récurrents (RNN) qui permettent de mieux gérer les dépendances à long terme dans les séquences de données. Les LSTM utilisent une cellule mémoire interne qui peut stocker et récupérer de l'information en utilisant des portes d'entrée, d'oubli et de sortie qui contrôlent le flux d'information dans la cellule [21]. Les LSTM ont été largement utilisés dans des tâches telles que la reconnaissance de la parole, la traduction automatique et la prédiction de séries temporelles.

1.3 Internet des objets

L'Internet des objets (IdO) ou internet of things (IoT) est une technologie en plein essor qui consiste à connecter des objets du quotidien au réseau Internet, dans cette section nous allons donner un aperçu sur l'objet connecté et quelques éléments qui le constitue.

1.3.1 Définition d'un objet connecté

C'est tout objet relié à internet et qui possède une capacité de recevoir, stocker, traiter, transmettre et d'échanger les données avec d'autres entités quelques soit physique ou numériques [22], possède aussi une adresse IP unique. Parmi les appareils intelligents, on trouve des électroménagers, des serrures, des caméras de sécurité

1.3.2 Définition de l'internet des objets

Est l'interconnexion entre l'internet et les objets, elle représente une révolution de l'informatique et de la communication. Cette révolution est basée sur une évolution constante de l'Internet, des technologies, des logiciels et des protocoles de communication [23]. L'IOT peut être appliqué dans plusieurs domaines et secteurs, comme le domaine de la santé, de l'industrie, du Transport et de La Mobilité Intelligent, le domaine de la sécurité et la surveillance, E-commerce, le domaine industriel...etc.

1.3.3 Étapes de mise en place de l'IdO

Il existe 5 étapes pour la mise en place de l'IdO qui sont :

— **L'identification :** c'est l'identification de chaque élément connecté.

— **L'installation de capteurs :** Mise en place de dispositifs qui va nous rapprocher du monde réel.

— **La connexion des objets entre eux :** faire une connexion entre les objets pour échanger les données.

— **L'intégration :** dans cette phase les données vont être transmises d'une couche à une autre.

— **La connexion à un réseau :** Relier les objets et leurs données au monde informatique via un réseau internet

1.3.4 Cycle de vie d'un objet connecté

Chaque objet passe par un " cycle de vie d'objet ", cela représente toutes les phases de la création à l'expiration.

- **La fabrication :** Dans cette phase, l'objet est créé et les caractéristiques physiques sont consolidées. Attribué l'adresse physique et l'ensemble de dispositifs intégrés.
- **Bootstrap :** Dans cette phase les objets sont intégrés dans l'environnement fonctionnel configuré (attribuer l'adresse réseau, prise de rôle, définir différents paramètres) et démarrés.
- **Phase opérationnelle :** Dans cette phase, l'objet travaille conformément à sa spécification initiale, et développer des règles de comportement de compensation du réseau.
- **Maintenance :**Réparation physique ou logicielles en cas de panne d'un objet. L'objet sera réparé et redémarré à partir de boot-phase de cerclage.
- **Révocation (expiration) :** La phase finale du cycle de vie d'un objet est la révocation lorsqu'il est définitivement hors service ou considéré comme obsolète.

1.3.5 Domaines d'application

L'iot est utilisé quasiment dans tous les domaines que nous côtoyons au quotidien. Nous en citons :

- **Le Transport :** L'IoT est utilisé dans le domaine du transport pour surveiller les véhicules, pour suivre les itinéraires, pour améliorer la sécurité routière, pour améliorer la maintenance des véhicules et pour optimiser la logistique.
- **L'industrie :** L'IoT est utilisé dans l'industrie pour surveiller et optimiser les processus de fabrication, pour suivre la maintenance des équipements, pour surveiller les niveaux de stock, pour améliorer la chaîne logistique, pour optimiser la production et pour améliorer la sécurité sur les sites industriels.
- **La santé :** L'IoT est utilisé dans le domaine de la santé pour surveiller la santé des patients à distance, pour suivre la prise de médicaments, pour surveiller les signes vitaux et pour améliorer la qualité des soins.
- **L'agriculture :** L'IoT est utilisé dans l'agriculture pour surveiller les cultures, pour surveiller les animaux, pour optimiser les niveaux d'irrigation et pour améliorer la production.
- **La domotique :** L'IoT est utilisé dans les maisons intelligentes pour contrôler les appareils domestiques, pour surveiller la sécurité, pour surveiller les niveaux d'énergie et pour améliorer le confort des occupants.

La figure suivante représente un schéma récapitulatif des différents domaines dont lesquels l'internet des objets peut être appliquée :

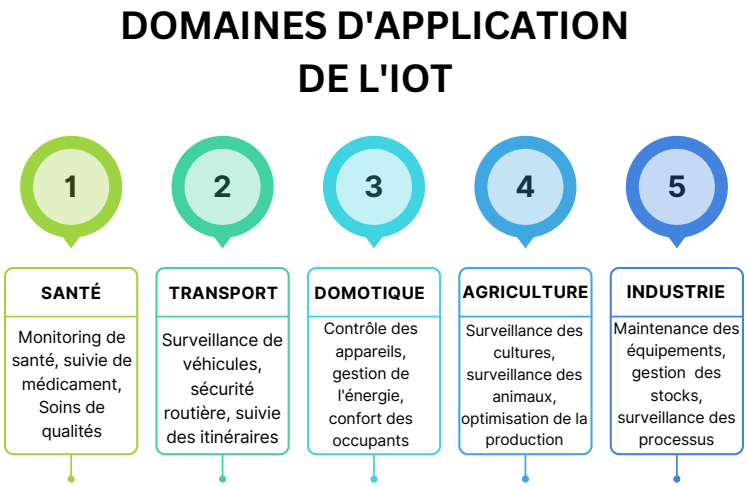

FIGURE 1.15 – Les domaines d'application de l'iot

1.3.6 Application de l'apprentissage automatique pour l'IdO

Aujourd'hui, de nombreux algorithmes ML sont implémentés dans l'IdO. Ces applications ML dépendent fortement du domaine appliqué. Plusieurs raisons expliquent l'impact de ML sur IdO. Mais que se passe-t-il en premier si l'IdO est mis en œuvre sans ML ?
L'internet des objets doit faire face aux conséquences suivantes lorsqu'il n'est mis en œuvre que sans ML. Cela comprend l'intégration de données provenant de plusieurs sources, la gestion des appareils, le contrôle de version des applications. L'IdO trait la mise en réseau d'appareils dans le but principal de partager des données. Ces données sont la raison pour laquelle le ML est puissant et améliore l'efficacité de l'IdO. Le ML contribue à IdO principalement en analysant les données et en prédisant les événements futurs, en transformant les données brutes en une forme compréhensible par l'homme, des systèmes de recommandation en temps réel et la maintenance des appareils (IoT). Analysant intelligemment les mégadonnées générées par des milliards de ces appareils, internet des objets a des applications dans de multiples domaines déjà cité avant [24].

Conclusion

Le Machine Learning (apprentissage automatique) et l'Internet des objets (IoT) sont deux domaines de la technologie qui ont connu une croissance exponentielle ces dernières années. Le Machine Learning consiste à enseigner à un système informatique à apprendre des modèles à partir de données, tandis que l'IoT fait référence à la connectivité des objets du quotidien à Internet, permettant ainsi de collecter et de partager des données. L'utilisation de l'IoT est de plus en plus répandue dans de nombreux secteurs, tels que la santé, l'agriculture, l'industrie, la logistique et la sécurité. Cependant, cette connectivité accrue peut également exposer les systèmes à des cyberattaques et à des vulnérabilités de sécurité. Dans ce chapitre, nous allons explorer la manière dont le Machine Learning peut être appliqué à la sécurité de l'IoT, notamment pour détecter les anomalies, prévenir les fraudes, améliorer la sécurité physique et des réseaux, et détecter les menaces potentielles. Nous examinerons également les défis et les

opportunités associés à l'utilisation du Machine Learning dans ce contexte, et les implications pour l'avenir de la sécurité de l'IoT. En fin de compte, le machine learning est une technologie en constante évolution qui continuera de jouer un rôle important dans la sécurité informatique. Les avancées technologiques et la collaboration entre les experts en sécurité et les développeurs de machine learning sont essentielles pour assurer une sécurité efficace et continue dans notre monde de plus en plus connecté.

Chapitre 2

Travaux connexes

Introduction

Avec l'expansion rapide de l'IoT, la sécurité de ces dispositifs est devenue une préoccupation majeure pour de nombreuses organisations. Les botnets sont l'une des menaces les plus importantes dans ce domaine, car ils peuvent compromettre la sécurité des dispositifs IoT et causer de graves dommages. Pour lutter contre ce problème, de nombreuses approches de machine learning et de deep learning ont été proposées pour la détection d'attaques basées sur les botnets. Dans cette étude, nous passerons en revue les différentes approches utilisées pour la détection des attaques de botnets dans l'IoT en utilisant des techniques de machine learning et de deep learning.

2.1 Littérature des propositions pour la détection des attaques de botnets

Dans cette section nous allons présenter les différentes méthodes utilisées dans la littérature pour la lutte contre les botnets en utilisant les techniques de machine et deep leraning, la figure suivante montre la classification suivie pour définir les différents travaux :

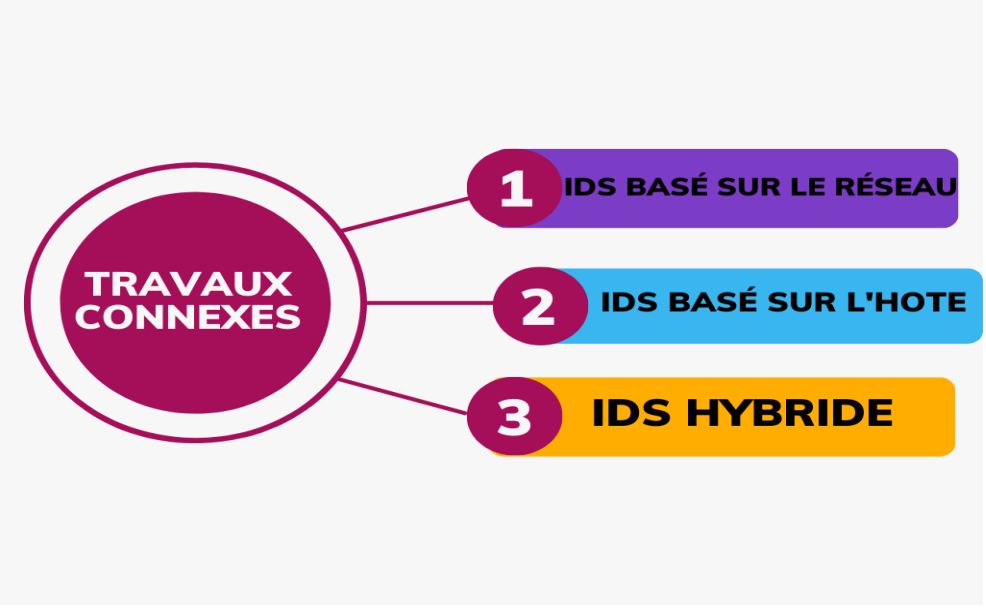

FIGURE 2.1 – Méthodes de défense contre les botnets en utilisant le machine et le deep learning

2.1.1 Solutions basées sur NIDS (Network Intrusion Detection Systems)

Satish Pokhrel et al. [25], ont utilisé un ensemble d'algorithmes d'apprentissage automatique, à savoir K plus proche voisin (KNN), Naïve Bayes et le réseau de neurones artificiels à perception multicouche (MLP ANN), pour développer un modèle visant à détecter et atténuer les attaques DDoS basées sur des botnets dans un réseau IoT. Pour ce faire, les auteurs ont utilisé un ensemble de données Bot-IoT. Ils ont combiné les algorithmes d'apprentissage automatique avec l'ingénierie des caractéristiques et la technologie de suréchantillonnage des minorités synthétiques (SMOTE). Enfin, ils ont comparé les performances des trois algorithmes sur le jeu de données de déséquilibre de classe et le jeu de données d'équilibre de classe. La sélection du meilleur algorithme a été basée sur le pourcentage de précision et l'aire sous la courbe des caractéristiques de fonctionnement du récepteur (ROC AUC).Les résultats obtenus ont montré que l'algorithme KNN était le plus efficace dans la détection des botnets, avec une précision allant jusqu'à 99,35 % et une ROC AUC de 0,995. La méthode proposée par les auteurs offre une solution efficace pour détecter et atténuer les attaques DDoS basées sur des botnets dans un réseau IoT. Elle permet de protéger les dispositifs IoT contre les attaques malveillantes et de garantir leur sécurité et leur intégrité.

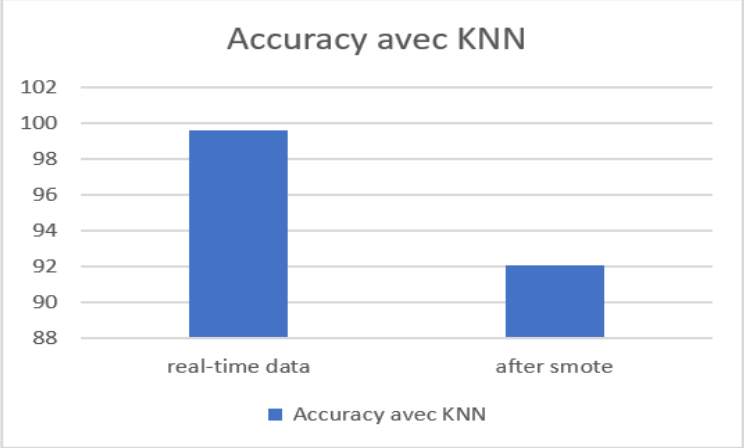

FIGURE 2.2 – Résultat de l'accuracy en utilisant KNN [25]

Arvind Prasad et Shalini Chandra [26], proposent une solution multimode basée sur le vote pour contrer les attaques DDoS volumétriques menées par des appareils IoT infectés. La solution VMFCVD repose sur trois modes de détection : la détection rapide (FDM), la détection rapide défensive (DFDM) et la haute précision (HAM). Pour évaluer leur solution, les auteurs ont utilisé 12 ensembles de données étiquetés contenant à la fois du trafic bénin et malveillant. Les auteurs ont divisé leur travail en plusieurs sections. Tout d'abord, ils ont comparé les trois modes de VMFCVD entre eux. Ensuite, ils ont comparé les performances de VMFCVD avec celles de différents modèles de machine learning, notamment le classificateur AdaBoost, le classificateur d'ensachage, le classificateur boostant le gradient, KNN et la forêt aléatoire La mesure de performance utilisée est la précision. Les résultats obtenus montrent que VMFCVD surpasse les autres études en termes de précision de classification, atteignant 99,9 %, avec une réduction de dimension des données de 98 %. Cette solution multimode permet donc de détecter efficacement les attaques DDoS volumétriques et de réduire considérablement les fausses alarmes.

Santha devi D [27], présente une solution pour améliorer la détection de Botnets dans les données Netflow en utilisant deux modèles d'apprentissage automatique, Ada-Boost et Stochastic Gradient Descent classification (SGDC). Les modèles sont entraînés et testés sur le dataset CTU-13, et leur précision de détection de botnet est comparée dans trois scénarios différents. Le dossier Netflow est divisé en fenêtres temporelles, et chaque fenêtre est étiquetée avec des prédictions. La mesure de performance prise en compte est la précision. Les résultats montrent que l'utilisation d'une méthode d'ensemble, combinant deux modèles de classification, peut améliorer considérablement la précision de détection des botnets dans les données Netflow, atteignant une précision de 98,31 % pour l'attaque IRC et une précision de 97,25 % pour les attaques DDoS. Cette approche peut donc être considérée comme une amélioration par rapport aux méthodes de détection existantes qui ont des taux de précision inférieurs.

Nookala Venu et al. [28], ont proposé une méthode de détection plus précise des attaques de botnet en utilisant la régression logistique et le comptage de fréquence. Pour entraîner leur modèle, ils ont utilisé trois ensembles de données : CICIDS2017, CTU-13 et IoT-23, afin de former quatre algorithmes populaires de machine learning pour détecter les attaques de botnet et de DDoS. Les modèles utilisés étaient la régression logistique (LR), KNN, forêt aléatoire (RF) et Naïve Bayes (NB). Les résultats obtenus à partir des trois ensembles de données ont montré que la méthode de forêt aléatoire a connu un succès remarquable, avec une précision de détection de 100 %. Cette solution pourrait donc aider à améliorer la détection des attaques de botnet dans l'Internet des objets, en utilisant une méthode de régression logistique et de comptage de fréquence plus précise que celles existantes.

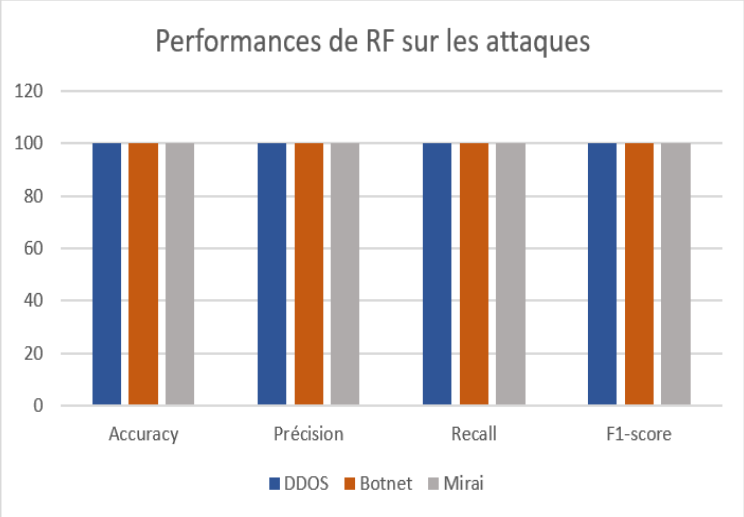

FIGURE 2.3 – Graphe qui représente l'accuracy avec le modèle RF [28]

Abhishek Raghuvanshi et al. [29], dans cet article établissent un cadre pour la détection et la classification des intrusions dans les réseaux IOT utilisés en agriculture. Ils ont utilisé le dataset NSL-KDD et pour l'extraction des caractéristiques c'est à l'aide d'analyse en composant principale ; SVM, la régression linière et les forets aléatoires pour la classification. Pour l'évaluation ils se sont basés sur l'exactitude, la précision et le rappel. Le jeu NSL-KDD est utilisé pour la détection d'anomalies, cette version ne contient pas de redondances, ni de duplication d'enregistrements. Pour y procédé les auteurs ont utilisé que 20 % des données d'apprentissage pour générer les règles de décision. Les résultats expérimentaux ont montré que cette méthode atteint une exactitude de 98 %, une précision de 99 % et un rappel de 96 %. Tenant compte des

résultats trouvés, cette méthode est plus précise que les approches précédentes et peut fournir une meilleure détection des attaques de botnet dans les réseaux IoT.

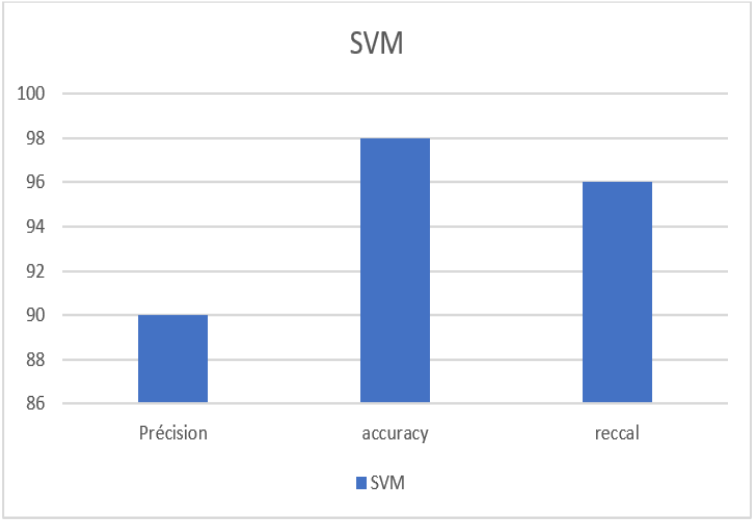

FIGURE 2.4 – Performance obtenues avec SVM [29]

2.1.2 Solutions basées sur HIDS (Host Intrusion Detection Systems)

Daniel GV Goncalves et al. [30], ont proposé dans cet article, une solution de détection d'intrusion basée sur l'hôte (HIDS) pour protéger les composants du réseau IoT backbone comprenant les commutateurs et les routeurs conventionnels, et non les dispositifs IoT. Cette solution utilise des techniques de vérification de sécurité conventionnelles et une interaction avec un contrôleur pour coordonner les actions de détection d'intrusion relatives aux attaques DDoS distribuées dans l'ensemble de l'instance IoT. Le HIDS distribué a été évalué dans un environnement contrôlé représentant des instances de réseaux IoT. l'article met en avant la capacité de cette solution à protéger les composants des réseaux IoT backbone contre les attaques DDoS distribuées tout en permettant une coordination efficace de la détection d'intrusion.

Vitor Hugo Bezerra et al. [31], dans cet article présentent une méthode de détection de botnets dans les appareils IoT en utilisant une approche de classification à une classe. Les auteurs ont construit une machine à vecteurs de support (SVM) à une classe en utilisant des fonctionnalités telles que l'utilisation du CPU et de la mémoire pour détecter les activités malveillantes. Pour évaluer les performances de leur approche, les auteurs ont effectué des tests dans un réseau contrôlé. Ils ont utilisé trois paramètres légitimes différents et sept botnets IoT pour simuler des scénarios d'attaques. Les résultats ont montré que le système proposé était efficace pour détecter différents botnets avec une consommation de ressources relativement faible. Comparé aux méthodes existantes, cette approche présente plusieurs avantages. Tout d'abord, elle se base sur des caractéristiques spécifiques aux hôtes IoT, telles que l'utilisation du CPU et de la mémoire, ce qui lui permet de détecter les activités malveillantes de manière précise et ciblée. De plus, en utilisant une classification à une classe, elle ne nécessite pas de données d'apprentissage sur les botnets spécifiques, ce qui la rend plus flexible et adaptable à différentes situations.

Dataset	Accuracy	Precision	Recall	F1-score	Specificity	AUC
MC_I1	0.9847	0.9597	0.9648	0.9622	0.9898	0.9773
MC_I2	0.9744	0.9627	0.9080	0.9345	0.9911	0.9495
MC_I3	0.9909	0.9659	0.9900	0.9778	0.9911	0.9906
SC_I1	0.9928	0.9654	1	0.9823	0.9909	0.9954
SC_I2	0.9926	0.9646	1	0.9819	0.9908	0.9954
SC_I3	0.9449	0.9596	0.7571	0.8463	0.9919	0.8745
ST_I1	0.9665	0.9459	0.8843	0.9140	0.9872	0.9358
ST_I2	0.9931	0.9671	1	0.9832	0.9913	0.9956
ST_I3	0.9933	0.9678	1	0.9836	0.9916	0.9958

FIGURE 2.5 – Meilleurs résultats pour l'approche proposée [31]

2.1.3 Solutions basées sur IDS Hybride :

Shreehar Joshi et al. [32], dans cet article ont pour objectif de trouver un classificateur efficace pour la détection des trafics d'anomalies. Pour se faire ils ont appliqué 4 algorithmes d'apprentissage automatique dont les forets aléatoires ; les arbres de décision, les machines à vecteur de support (SVM) et d'un classificateur d'arbre supplémentaire (extra tree classifier). Le dataset utilisé est le NBa-Iot, les évaluations sont effectuées avec des données de trafic extraites de 4 appareils infectés par deux sortes de Bot à savoir : "Mirai" et "BASHLITE". Les données des différentes classes d'attaques utilisées dans cette étude sont déséquilibrées. Pour évaluer les performances des différents algorithmes et procéder à la sélection du meilleur les auteurs ont pris en considération la mesure F1 score. A partir des nombreux tests effectués ils ont finalement déduit que les forets aléatoires est l'algorithme de machie learning le plus efficace pour la détection des botnets dans l'iot prenant en considération à la fois le F1 score et le temps d'exécution. De plus, la méthode proposée par les auteurs peut être appliquée à différents types de botnets, ce qui augmente sa polyvalence et son utilité dans la lutte contre les attaques dans l'IoT.

Miloud Bagaa et al. [33], proposent une solution dans le cadre de l'utilisation de machine learning pour améliorer la sécurité de l'IOT. Pour se faire ils ont combiné un réseau défini par logiciel (SDN), la visualisation des fonctions réseaux (NFV) et les techniques de ML. D'abord ils ont étudié les menaces les plus pertinentes dans l'iot, ensuite ils ont combiné un agent de surveillance et un agent de rédaction qui utilisent les modèles de ML, mais aussi un système de détection d'intrusion basé sur l'anomalie dans l'iot. Pour évaluer l'IDS, ils ont mis en place un scénario de bâtiment intelligent en utilisant l'algorithme SVM, la précision de détection des anomalies est de 99,7 %. Le cadre proposé fournit une approche structurée pour appliquer l'apprentissage automatique dans la sécurité de l'IOT, et il souligne l'importance de la surveillance et de l'évaluation continues des mesures de sécurité pour garantir leur efficacité. De plus, la combinaison de techniques de SDN, de NFV et de ML offre une approche plus complète et intégrée pour la sécurité de l'IoT, ce qui en fait une solution plus avancée.

Fereshteh Abbasi et al. [34], ont étudié en profondeur des méthodes de ML pour la détection d'intrusion dans l'IOT, ils ont proposé deux méthodes d'extraction et de classification des caractéristique, la première à l'aide de la régression logistique (LR), et la seconde consiste à utiliser les réseaux de neurones artificiels (ANN). Pour évaluer les performances ils ont utilisé 6 appareils de l'ensemble de données N-BaioT et 9 appareils IOT et de plusieurs attaques utilisées. Dans cet article ils ont abordé les attaques bashlite et mirai, et ils ont utilisé la régres-

sion logistique pour la sélection des caractéristiques, et la classification en utilisant ANN. Pour évaluer l'efficacité des méthodes proposées ils ont utilisé la précision, la courbe ROC, TPR, FPR, l'accuracy, le rappel et le F1 score. Les resultats de simulation par rapport à certaines méthodes montre que l'utilisation de la régression logistique est plus efficace avec une précision de classification supérieur à 90 %, et pour les réseaux de neurones les meilleures resultats sont obtenus pour ceux avec 3 couches.

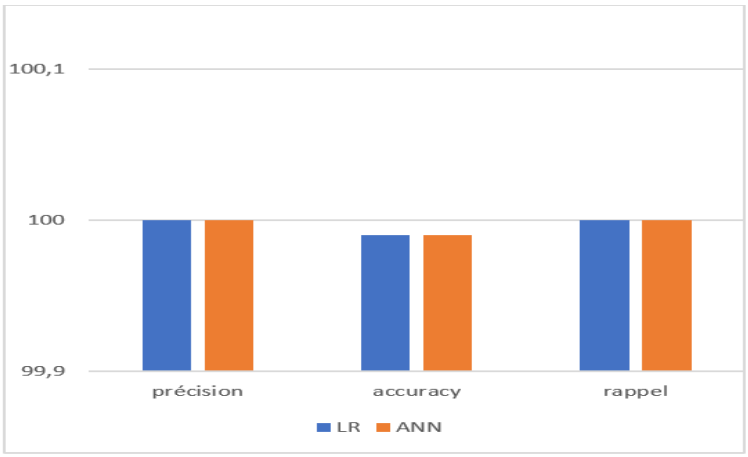

FIGURE 2.6 – performances des méthodes LR et KNN [34]

Parul Agarwal et al. [35], visent à explorer une méthode de détection d'anomalies en utilisant les techniques de Machine Learning telles que SVM, ANN, KNN, LR, DT et RF pour neutraliser les menaces et renforcer la cybersécurité d'une ville intelligente. Les auteurs ont examiné le rôle des méthodes d'ensembles comme le bagging, boosting et le Stacking pour avoir une couche supplémentaire de sécurité. Pour cela ils ont utilisé les deux datasets UNSW-BC15 et CICIDS2017 et les mesures de performance : l'exactitude, la précision, le rappel et le F1-score. Ils ont proposé trois couches pour la ville intelligente : couche brouillard, couche liaison et la couche terminale qui doit être en communication avec les appareils IOT. D'après leurs resultats de la courbe ROC pour les déférentes techniques ils ont obtenu de meilleures performances avec les deux algorithme, arbre de décision (DT) et les forets aléatoires (RF), et la méthode d'ensemble Stacking est meilleure que les autres, la valeur de F1-score de leur méthode est supérieure à ceux qui existe déjà.

Abdellatif El Ghazi et Ait Moulay Rachid [36], proposent un système hybride de détection d'intrusion qui utilise à la fois le cloud et le brouillard pour surveiller les communications et détecter les attaques en temps réel afin de garantir la sécurité des appareils IOT. Ils ont utilisé les algorithmes SVM, ANN et l'analyseur PCA. L'objectif d'IDS proposé dans cet article est de minimiser le taux des faux positifs et de maximiser la détection des attaques zéro Day pour l'IDS par signature avec l'algorithme SVM ; après avoir collecter les signatures un autre système de détection d'anomalies basé sur les réseaux de neurone va améliorer le taux de la détection en ligne. Ils ont essayé de combiner un vrai positif élevé d'utilisation abusive de l'IDS et d'augmenter le taux de détection en obtenant une meilleure qualité des données. Cette approche utilise une combinaison de techniques de ML et d'ensembles pour améliorer la détection d'anomalies dans les IDS pour les villes intelligentes, offrant ainsi une meilleure sécurité globale.

Rakhi Sarathe et Sumit Sharma [37], proposent un système de détection d'intrusion réseau IOT qui détecte la session attaque et celle normale, les auteurs ont utilisé l'algorithme génétique d'optimisation de la flamme papillon pour la sélection des caractéristiques qui permet d'identifier les sessions. Les étapes du modèle proposé Moth Flame based IOT Network Security (MFIOTNS) sont résumées comme suit : le nettoyage des data-sets, optimisation des fonctionnalités, l'algorithme d'optimisation de flamme Moth, générer des flammes de papillon, fonction de remise en forme, mettre à jour la position de la flamme papillon, croisement et en dernière étape la fonction de filtre. Pour tester leur resultats ils ont utilisé la mesure de précision. D'après les resultats, le modèle proposé a amélioré les performances de détection d'intrusion par rapport aux modèles existants de 5,004 % en termes de précision, et l'efficacité du travail en optimisant diverses valeurs de paramètres d'évaluation.

2.1.4 Analyse comparative

Dans cette sous-section nous allons faire une comparaison entre les différentes méthodes que nous avons décrit ci-dessus.

2.1.4.1 Schéma récapitulatif :

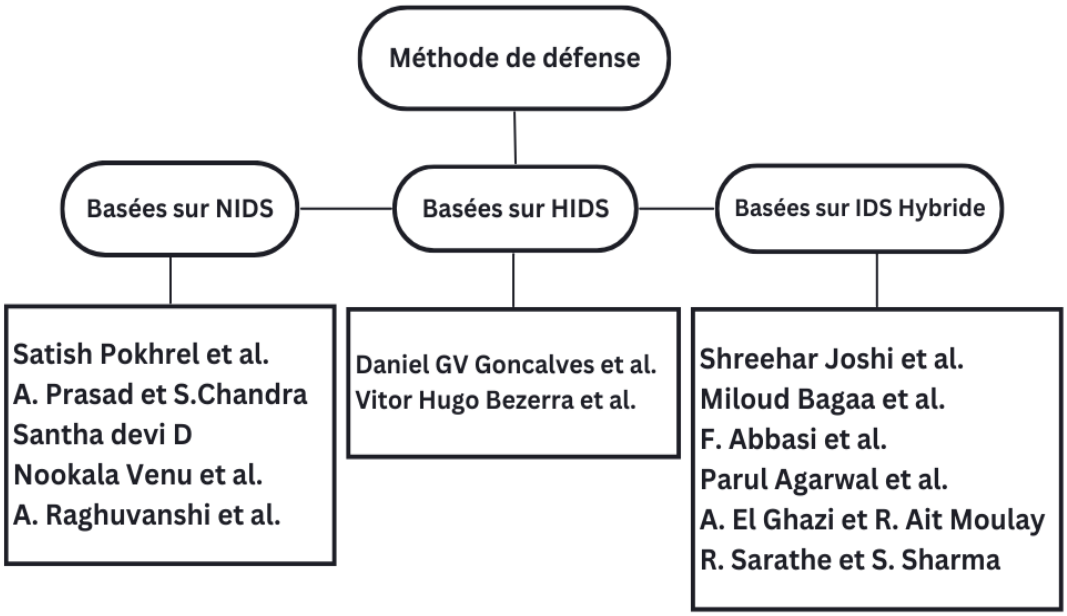

FIGURE 2.7 – Méthodes de défense avec NIDS, HIDS et IDS Hybride

2.1.4.2 Tableau Comparatif :

TABLE 2.1: Comparaison entre les différentes méthodes

Classes	Travaux	Critères			
		Techniques	Dataset	Mesures de performances	Année
NIDS	Satish Pokhrel et al.[25]	KNN Naive Bayes MLP ANN	Bot-IoT	Les taux de l'accuracy sont de : 99,6% avec D1 92,1% avec D2	2021
	A. Prasad et S. Chandra [26]	VMFCVD avec les trois modes de détection : FDM, DFDM et HAM	CICIDS2017 CSE-CIC-IDS2018 CICDDoS2019 DoHBrw2020 NBaIoT2018 UNSW NB15 UNSW2018	Une précision de 99,9% avec une réduction de dimension de données de 98%.	2022
	Santha devi D [27]	Ada-Boost SGDC	CTU-13	La meilleure précision est de 98,39% pour les SPAM avec Ada-Boost. 99,31% d'accuracy et F1 Score de 99,29% pour l'attaque DDoS avec Ada-Boost	2020
	Nookala Venu et al.[28]	RF, NB KNN, LR	CICIDS2017 CTU-13 IoT-23	100% d' Accuracy, précision, F1 Score et rappel avec les trois datasets pour RF	2022
	A. Raghuvanshi et al.[29]	SVM, RF, LR	NSL-KDD	Meilleur performances avec SVM, le taux de son accuracy, précision et rappel son supérieur à 98%.	2022
	Daniel GV Gonçalves et al.[30]			Le HIDS distribué conçu est évalué dans un environnement contrôlé qui représente de manière réaliste les instances de réseaux IoT, même s'il s'agit d'un réseau local et isolé.	2019

Table 2.1 continued from previous page

HIDS	Vitor Hugo Bezerra et al.[31]	OSVM	MC_I1, MC_I2, MC_I3, SC_I1, SC_I2, SC_I3 ST_I1, ST_I2, ST_I3	Meilleur performances avec ST_I3 pour : Une accuracy de 99,3% Précision de 96,78% Rappel de 100% F1_Score de 98,36% AUC de 99,58%	2018
IDS Hybride	Shreehar Joshi et al.[32]	DT, RF, ETC SVM	N-BaIoT	Meilleur F1_Score avec RF est de 95%	2020
	Miloud Bagaa et al.[33]	SVM Ensemble DNN	NSL-KDD	Meilleur accuracy avec SVM pour un taux de 98%	2020
	F. Abbasi et al.[34]	LR, ANN	N-BaIoT	Une Précision de 100% Une Accuracy de 99,99% Un Rappel de 100%	2021
	Parul Agarwal et al.[35]	LR, SVM, KNN, ANN, RF, DT et les méthodes d'ensembles	CICIDS2017 UNSW-BC15	Meilleurs performances avec DT et RF. La méthode de stacking a montré de meilleures performances par rapport aux autres.	2023
	A. El Ghazi et R. Ait Moulay [36]	ANN et SVM	Les données sont capturées en direct depuis Internet à l'aide d'un Honeypot.	Une précision de 60% Un rappel de 50%	2020
	R. Sarathe et S. Sharma[37]	KNN MFIOTNS	IOT dataset	Le modèle proposé améliore la valeur de la précision de 5,004% et de l'accuracy de 6,25% par rapport aux modèles précédent.	2022

La majorité des articles mentionnés ci-dessus ont suggéré différentes méthodes d'apprentissage automatique et d'apprentissage profond pour identifier les attaques de Botnet. Les cinq premiers travaux [25], [26], [27], [28] et [29], ont proposé des approches basées sur NIDS mais en utilisant différentes techniques de ML et DeepL, cependant toutes ne s'avèrent pas très efficaces tenant compte des résultats obtenus lors de l'évaluation des performances, dans [25] et [29] les performances des modèles sont plutôt faible comparer à d'autres solutions proposées. Dans [30] et [31] les auteurs ont proposé des solutions basées sur l'hôte, la solution proposée dans [31] n'a utilisé que deux types d'appareils IoT et basé sur un nombre très limité de fonctionnalités des hôtes relatives à ces appareils. Bien que les auteurs ont démontré que leurs solutions étaient efficace contre les attaques de botnets, mais elles ne sont pas plus fiables que celles basées sur le réseaux compte tenue des ressources limitées et du volume des appareils Iot. Dans les six derniers articles les solutions sont hybrides, les performances des modèles proposés varient entre 50% et 100% pour le rappel, la précision, le F1 score et l'accuracy, ce qui est parfaitement acceptable, néanmoins ce type de modèle à besoin de plus de temps de calcul et de ressources pour l'entraînement et l'évaluation mais il est aussi exposé au risque de sur-ajustement (overfitting).

Conclusion

En conclusion, nous avons examiné plusieurs approches utilisées pour la détection d'attaques basées sur les botnets dans l'IoT par des approches de machine et de deep learning. Nous avons constaté que ces approches sont prometteuses et peuvent fournir une détection efficace des attaques basées sur les botnets dans l'IoT. Cependant, il y a encore des défis à relever dans ce domaine, tels que l'adaptation des modèles à de nouveaux types d'attaques de botnets dans l'IoT et l'amélioration de la performance de détection en temps réel. Nous espérons que cette étude servira de base pour de futures recherches dans le domaine de la détection des botnets dans l'IoT et aidera à renforcer la sécurité de l'IoT.

Chapitre 3

Contribution, résultat et discussion

Introduction

Les systèmes de détection d'intrusion basés sur l'apprentissage automatique et profond ont suscité de nombreuses études de recherche. Dans le chapitre précédent nous avons examiné les diverses techniques de ML et DeepL qui ont été appliquées avec succès à la détection d'intrusion en utilisant différents jeux de donnés. Les performances des IDS basés sur ML et DeepL sont fortement influencées par le choix du Dataset à utilisés. Ce dernier chapitre contient la réalisation et l'implémentation de notre solution. Nous décrivons en détail le protocole que nous avons suivi pour construire notre modèle IDS, en mettant l'accent sur les étapes clés telles que la préparation des données, la standardisation et la normalisation, l'entraînement des modèles, et la validation des performances.

Dans la première partie de ce chapitre, nous expliquons en détail la préparation des données pour l'apprentissage.

Ensuite, nous abordons la standardisation et la normalisation des données, qui sont des étapes essentielles avant de former nos modèles IDS.

Nous poursuivons avec la description de l'entraînement des modèles IDS basés sur l'apprentissage automatique. Pour cela nous avons utilisé plusieurs techniques d'apprentissage automatique et une technique de l'apprentissage profond. Chacun de ces modèles a été formé sur des ensembles de données normalisées ou standardisées selon des configurations spécifiques.

Enfin, nous abordons la validation des modèles et l'évaluation des performances, pour évaluer la capacité de nos modèles à prédire les classes d'intrusion, dans notre étude nous avons pris une marge d'erreur équivalente à 10^{-2}.

Ce dernier chapitre offre un aperçu complet du processus de réalisation et d'implémentation de notre solution IDS basée sur l'apprentissage automatique pour la détection d'intrusion. Les détails spécifiques de chaque étape et des modèles utilisés sont disponibles dans les sections suivantes.

3.1 Environnement de développement

3.1.1 Matériel utilisé

Dans ce qui suit, nous allons présenter les configurations de la machine sur laquelle notre système a été développé :

TABLE 3.1 – Caractéristiques du matériel

Matériels	Caractéristiques
PC	Processeur : Intel (R) Core (TM) i5-4310M CPU @ 2.70 GHz. Mémoire (RAM) : 8,00 Go. Disque Dur : 512 Go HDD, 120 Go SSD. Système d'exploitation : Windows 10 Professionnel

3.1.2 Langages de programmation et bibliothèque

Nous avons utilisé Jupyter notebooks dans l'environnement Anaconda, et différentes bibliothèques que nous allons présenté dans ci-dessous :

Anaconda : est une distribution gratuite et open source de Python et R, dediée au développement d'applications pour l'apprentissage automatique et la science des données. Elle fournit un environnement complet et préconfiguré avec de nombreuses bibliothèques et outils essentiels pour ces domaines [38].

Jupyter : a été créé en 2014 par une équipe de développeurs dirigée par Fernando Pérez, un physicien et informaticien.C'est un environnement de développement interactif largement utilisé pour l'exécution de code, la visualisation de données et l'analyse de données. Il permet de créer et de partager des documents appelés "notebooks" qui contiennent à la fois du code exécutable, des résultats, des visualisations, du texte explicatif et des éléments interactifs.L'un des principaux avantages de Jupyter est sa prise en charge de multiples langages de programmation, notamment Python, R, Julia et d'autres.En plus de l'environnement de développement local, Jupyter offre également des services de notebook en ligne tels que Jupyter Notebook (anciennement connu sous le nom d'IPython Notebook) et JupyterLab, qui permettent de créer, exécuter et partager des notebooks via un navigateur web [39].

Nous avons utilisé comme bibliothèque pour notre projet :

- **Pandas :** permet la manipulation et l'analyse des données et offre des structures de données et des opérations pour travailler avec des tableaux de chiffres.
- **Numpy :** est une bibliothèque Python pour le calcul numérique qui permet la manipulation rapide et efficace de tableaux multidimensionnels.
- **Scikit-Learn :** nous permet d'expérimenter différentes techniques et algorithmes d'apprentissage automatique et d'analyser les données prédéfinis rapidement et facilement.
- **Matplotlib :** est utilisée pour tracer et visualiser des données sous formes des graphiques.

3.1.3 Données utilisées pour l'expérimentation

Une des étapes les plus importantes pour évaluer et valider les approches de détection d'attaques basées sur l'apprentissage automatique en cybersécurité est le choix des datasets a utiliser. Cependat l'accès à ces données peut s'avérer difficile à obtenir en vue de leur confidentialité. Dans le tableau ci-dessous nous allons présenter quelques ensembles de données conçu pour la détection des cyberattaques [40] :

Table 3.2: Datasets publiés sur les cyberattaques

Dataset	Type d'attaque	Etiqueté	Année
NSL-KDD	Attaques réseaux	Oui	2009
UNSW-NB15	Attaques réseaux	Oui	2015
CICIDS2017	Attaques réseaux	Oui	2017
KDD Cup	Attaques réseaux	Oui	1999
CTU-13	Attaques réseaux	Oui	2011
CICDDoS2018	Attaques par DDoS	Oui	2018

Dans cette sous-section nous allons présenté le Dataset utilisé pour la réalisation de notre approche.

3.1.4 Le dataset Bot-Iot

Est un ensemble de données spécifiquement créé pour la détection des botnets dans l'internet des objets (iot). Il contient des enregistrements de trafic réseau générés par des appareils iot réels, tels que des caméras de sécurité, des thermostats, des imprimantes, etc. Ces enregistrements représentent un scénario d'attaque où certains de ces appareils ont été infectés par des logiciels malveillants et sont contrôlés par des botnets.

Table 3.3: Caractéristiques du dataset utilisé

Dataset	Taille	Scénarios d'attaques	Caractéristiques	Etiquetage
Bot-IoT	1,083,744 enregistrements.	Comprend des données de trafic réseau générées à partir de diverses attaques de botnet, telles que l'infection, le controle et la communication entre les noeuds compromis.	Incluent des informations sur le trafic réseau, telles que : les adresses IP source et destination, les ports, les protocoles, les durées de connexion, les tailles des paquets, etc. Mais également des informations sur le comportement des botnet.	les enregistrements sont marqués comme étant soit liés à des activités malveillantes de botnet, soit à du trafic normal.

Deux classes principales figurent dans ce dataset, les accès de type Bénin (Traffic légitime) et les Attaques. Les types d'attaques que contient le dataset sont résumés dans le tableau ci-dessous :

TABLE 3.4: Types d'attaques dans le dataset Bot-Iot

Catégorie	Type d'attaque	Nombre de flux	Entrainement	Test
Bénin	Bénin	9543	7634	1909
Collecte d'information	Exploration de services	1463364	117069	29267
	Empreinte d'OS	358275	28662	7166
Attaque DDoS	DDoS TCP	19547603	1563808	390952
	DDoS UDP	18965106	1517208	379302
	DDoS HTTP	19771	1582	395
Attaque Dos	DoS TCP	12315997	985280	246320
	DoS UDP	20659491	1652759	413190
	DoS HTTP	29706	2376	594
Vol d'information	Keylogging	1469	1175	294
	vol de données	118	94	24

La figure 3.1 met en évidence le déséquilibre entre le trafic bénin et le trafic malveillant observé dans le jeu de données BoT-IoT. On observe une tendance exponentielle lors de l'évaluation de la répartition des échantillons par classe dans le jeu de données BoT. Cette tendance indique que le pourcentage d'échantillons malveillants est plus élevé que celui des échantillons bénins. Les classes majoritaires dans le jeu de données BoT-IoT correspondent aux types d'attaques, tandis que le trafic normal fait partie des classes minoritaires.

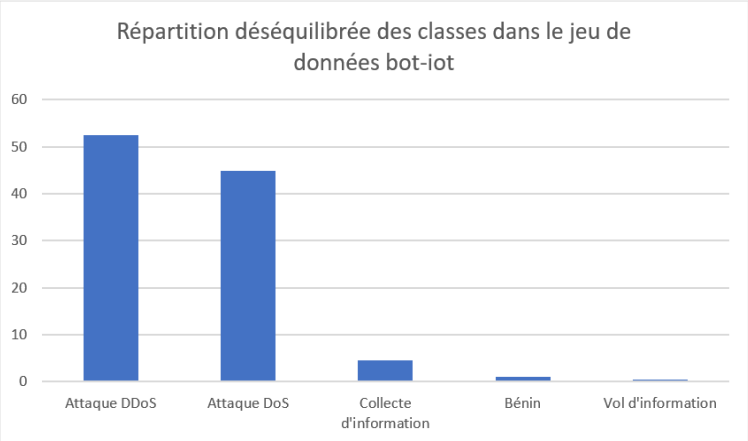

FIGURE 3.1 – Répartition déséquilibrée des classes dans le jeu de données bot-iot

3.2 Architecture et emplacement

Dans cette section nous décrivons l'architecture de l'IdO au niveau des couches OSI, ensuite nous aborderons les possibilités de l'emplacement de l'IDS dans le contexte de l'internet des objets :

3.2.1 L'IdO dans l'architecture en couches

Dans le modèle OSI (Open Systems Interconnection) qui décrit les différentes couches de communication dans un réseau informatique, l'IoT se situe généralement dans la couche de l'application (7ème couche). C'est à ce niveau que les applications et les services IoT interagissent avec les données collectées par les objets connectés.

Cependant, il est important de noter que l'IoT peut également nécessiter des interactions avec d'autres couches du modèle OSI, telles que la couche réseau (3ème couche) pour la transmission des données sur le réseau, la couche transport (4ème couche) pour le contrôle de la qualité de service, ou même la couche physique (1ère couche) pour la connectivité sans fil ou la communication filaire.

Une solution IdO ou IoT est composée de plusieurs couches interconnectées qui permettent de relier le monde physique des appareils au monde digital. Tous les cas d'usage n'adoptent pas une architecture unique. En revanche, ils font tous (ou presque) appel aux quatre mêmes couches principales représentées dans le schéma ci-après.

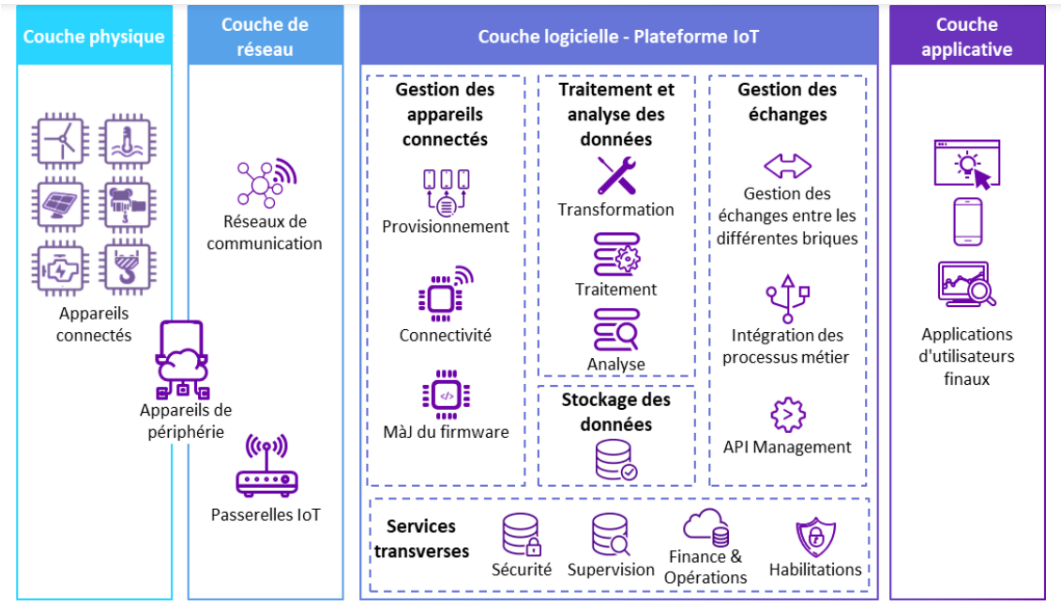

FIGURE 3.2 – Architecture d'une solution IoT
[41]

3.2.2 Positionnement de L'IDS dans l'IoT

L'emplacement d'un IDS (Système de Détection d'Intrusion) dans l'IoT dépend de plusieurs facteurs, tels que les objectifs de sécurité, les contraintes de performance, les caractéristiques du réseau IoT et les types de menaces ciblées. Voici quelques options d'emplacement possibles pour un IDS dans l'IoT :

- **Emplacement au niveau du périphérique (End-point) :** L'IDS est directement intégré dans les périphériques IoT pour surveiller les activités et détecter les comportements anormaux. Cela permet une détection précoce des intrusions au niveau des périphériques individuels, mais peut nécessiter des ressources matérielles et logicielles suffisantes.
- **Emplacement au niveau de la passerelle (Gateway) :** L'IDS est déployé sur une passerelle IoT, qui agit comme un point d'entrée centralisé pour les périphériques connectés. Cela permet de surveiller et d'analyser le trafic entrant et sortant de l'IoT, offrant une visibilité et une protection globales.
- **Emplacement au niveau du réseau :** L'IDS est positionné au sein du réseau IoT, surveillant le trafic entre les périphériques, les passerelles et les serveurs IoT. Cela permet de détecter les activités suspectes ou les anomalies au niveau du réseau, mais peut nécessiter une infrastructure réseau adaptée pour l'inspection du trafic.
- **Emplacement dans le cloud :** L'IDS est hébergé dans le cloud, où il analyse le trafic provenant de multiples sources IoT. Cela offre une évolutivité et une centralisation des

fonctions d'analyse, mais nécessite une connexion Internet fiable et peut soulever des préoccupations de confidentialité des données.

Il est important de noter que le choix de l'emplacement de l'IDS dans l'IoT dépendra des besoins spécifiques en matière de sécurité, de performance et de gestion du réseau. Une approche hybride combinant plusieurs emplacements peut également être envisagée pour une protection complète de l'IoT.

Dans ce qui suit une figure qui illustre une des possibilité de l'mplacement d'un IDS dans le contexte de l'IOT :

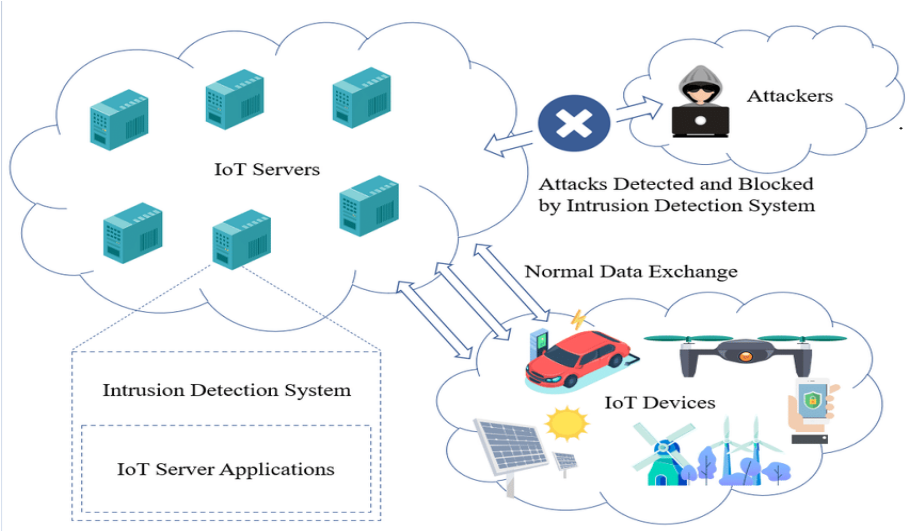

FIGURE 3.3 – Système de Détection d'Intrusion pour les Serveurs IoT [42]

3.3 Description du protocole de réalisation des modèles

Dans cette section, nous décrivons en détail le protocole que nous avons suivi pour construire notre modèle IDS basé sur l'apprentissage automatique afin de détecter les botnets dans le contexte de l'Internet des objets (IoT). Notre approche est résumée dans les figures 3.4 et 3.5, la première présentant les principales étapes de préparation de l'ensemble de données, tandis que la deuxième démontre les étapes de prétraitement, d'entraînement des modèles et de validation.

3.3.1 Préparation des données

La première étape pour préparer les ensembles de données pour l'apprentissage automatique consiste à nettoyer les données en supprimant les valeurs aberrantes, infinies et NaN (non numériques).

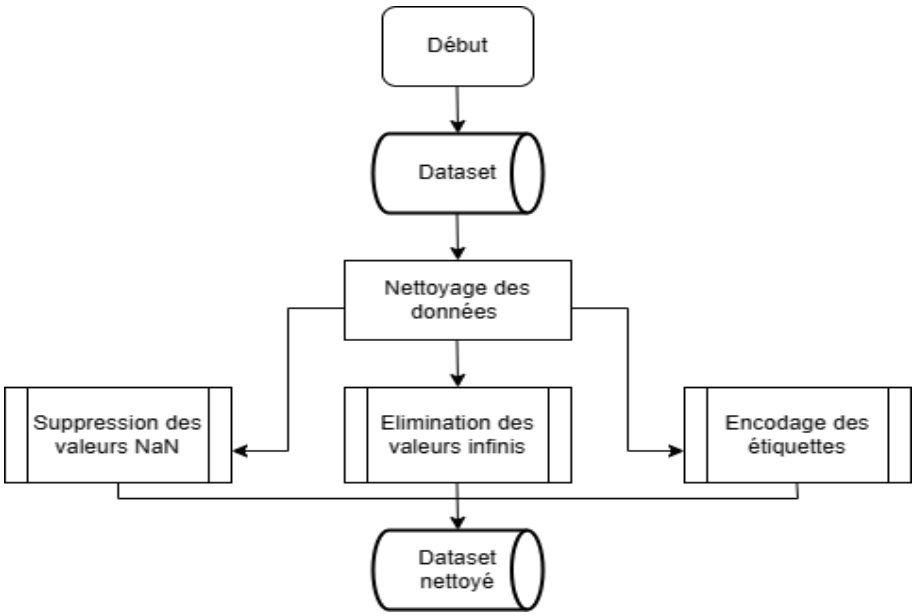

FIGURE 3.4 – Diagramme de flux de préparation des données [43]

3.3.2 Construction du modèle

Dans cette sous-section nous présentons un diagramme explicatif sur les étapes suivies pour la réalisation de notre implémentation :

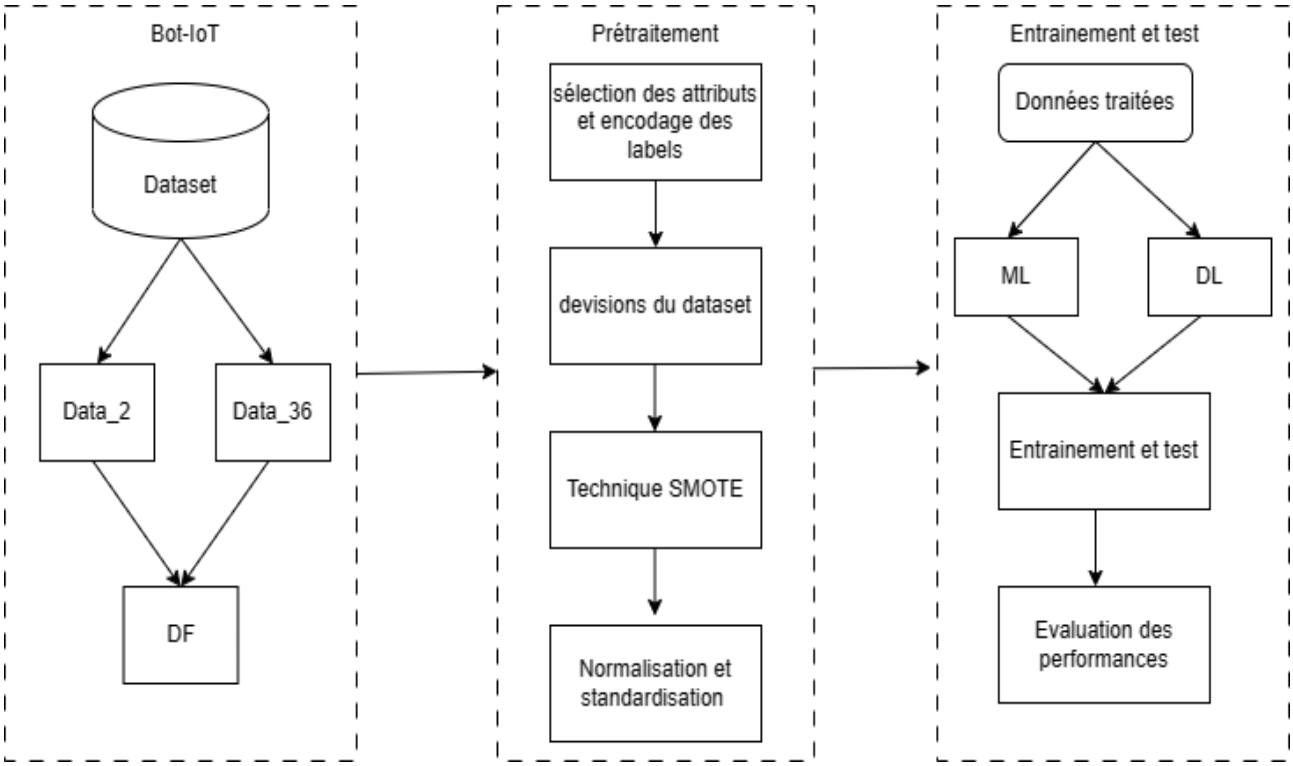

FIGURE 3.5 – Diagramme de flux de la construction du modèle IDS

3.3.3 Standardisation et normalisation

Comme le montre la figure ci-dessous l'échantillon a été divisé en ensembles d'entraînement et de test selon un ratio de 70% et 30% respectivement. Avant de former nos modèles, nous avons standardisé et normalisé les données. La standardisation a été réalisée en utilisant le score z, qui exprime les valeurs en fonction du nombre d'écart-types entre l'observation originale et la moyenne de la population. La normalisation a été effectuée en utilisant la méthode Min-Max pour mettre à l'échelle les données dans une plage de valeurs comprises entre 0 et 1, ce qui facilite l'entraînement de plusieurs modèles.

- **Standardisation :**

$$Z\text{-score} = \frac{X - \mu}{\sigma} \tag{3.1}$$

  Ou ***Mu*** est la population moyenne et ***Sigma*** est l'écart-type. Le Z-score exprime les valeurs en tant que nombre d'écart-types entre l'observation originale et la moyenne de la population.

- **Normalisation :**

$$\text{Min\_Max} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{3.2}$$

  Cette méthode est utilisée pour mettre à l'échelle l'intervalle des données entre des valeurs de 0 et 1, ce qui simplifie l'entraînement de plusieurs modèles.

3.3.4 Entraînement des modèles

Nous avons entraîné plusieurs modèles en utilisant différentes techniques d'apprentissage automatique. Les modèles ont été formés sur des ensembles de données normalisées avec la fonction Min-Max et standardisées avec le score z.

Modèles entraînés sur des données normalisées avec la fonction Min-Max :

- Régression logistique (LR).
- Analyse discriminante linéaire (LDA).
- Naive Bayes (NB).
- Perceptron multi-couches (MLP) : Nous avons entraîné des classifieurs MLP (Perceptron à plusieurs couches) avec plusieurs configurations, qui sont des combinaisons de :
  — Fonctions d'activation : Logistique, tangente hyperbolique, unité linéaire rectifiée.
  — Solveurs : Optimiseur à méthodes quasi-Newton (lbfgs), descente de gradient stochastique (sgd) et optimiseur basé sur le gradient stochastique (adam).
  — Couches cachées : Deux ou trois, avec un nombre maximal de nœuds cachés de 15.
  — Alpha (terme de régularisation L2) : 1e-5.
  — Taux d'apprentissage adaptatif : Cela maintient le taux d'apprentissage constant jusqu'à ce qu'il ne parvienne pas à réduire la perte d'entraînement entre deux époques, puis divise le taux d'apprentissage par 5.

Modèles entraînés sur des données standardisées avec le score z :

- K plus proches voisins (KNN).
- Arbres de décision (DT).

- Forêts aléatoires (RF).
- Machines à vecteurs de support (SVM) : Nous avons entraîné plusieurs modèles SVM en variant les fonctions de noyau (polynomial, gaussien, fonction de base radiale (RBF) et sigmoïde). La valeur du paramètre gamma a été fixée à l'inverse du nombre de fonctionnalités, tandis que le paramètre de régularisation C a été varié entre 0,1 et 2.

3.3.5 Validation des modèles et évaluation des performances

À la fin du processus, les modèles entraînés ont été testés sur l'ensemble de données de test afin d'évaluer leur capacité à prédire les classes. Pour évaluer nos modèles, nous avons utilisé les indicateurs de performance les plus pertinents et couramment utilisés, ces métriques sont calculées à l'aide de quatre mesures différentes, vrai positif (TP), vrai négatif (TN), faux positif (FP) et faux négatif (FN) :

— TP : Lorsqu'une instance anormale est identifiée comme telle, elle est correctement classée en tant que vrai positif (TP).
— FP : Lorsqu'une instance normale est incorrectement identifiée comme anormale, elle est erronément classée en tant que faux positif (FP).
— TN : Lorsqu'une instance normale est correctement identifiée comme normale, elle est acceptée en tant que vrai négatif (TN).
— FN : Lorsqu'une instance anormale est incorrectement identifiée comme normale, elle est erronément classée en tant que faux négatif (FN).

- **Accuracy :** exprime la portion de ce qui a été correctement prédit.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.3}$$

- **Précision :** évalue les cas positifs prédits par rapport à tout ce qui a été prédit comme positif, y compris les cas négatifs prédits comme positifs (faux positifs).

$$\text{Précision} = \frac{TP}{TP + FP} \tag{3.4}$$

- **Rappel :** mesure les positifs prédits par rapport à tous les cas réels positifs.

$$\text{Rappel} = \frac{TP}{TP + FN} \tag{3.5}$$

- **F1-score :** Il s'agit d'un paramètre global utilisé pour évaluer les modèles de classification binaire. Il est représenté par la moyenne harmonique du rappel et de la précision.

$$\text{F1-Score} = \frac{2 \cdot (\text{Rappel} \cdot \text{Précision})}{\text{Rappel} + \text{Précision}} \tag{3.6}$$

- **La courbe ROC (Receiver Operating Characteristic) :** : est un outil graphique utilisé pour évaluer et visualiser les performances d'un modèle de classification binaire. Elle représente la relation entre le taux de vrais positifs (sensibilité) et le taux de faux positifs

(1 - spécificité) à différents seuils de classification. La courbe est tracée en calculant ces taux à différents seuils et en les représentant sur un graphique. L'aire sous la courbe ROC (AUC-ROC) mesure la capacité du modèle à distinguer entre les classes prédites. Une AUC-ROC proche de 1 indique un modèle plus efficace.

- **Matrice de confusion :** est un outil utilisé pour évaluer les performances d'un modèle de classification en comparant les prédictions du modèle avec les classes réelles. Elle présente les résultats sous forme de tableau, où les prédictions correctes sont représentées sur la diagonale et les erreurs de classification sont réparties dans les autres cellules. Comme le représente le tableau suivant :

Table 3.5: Matrice de confusion

Classe prédite		1	0
Classe réelle	1	True Positive	False Negative
	0	False Positive	True Negative

3.4 Implémentation

Dans cette section, nous décrivons les détails de l'implémentation de notre méthode.

3.4.0.1 Importation des bibliothèques

Nous commencerons par importer les bibliothèques nécessaires que nous avions cité dans la section 3.2, comme suit :
• pandas (importé sous le nom pd) : pour la manipulation des données.
• numpy (importé sous le nom np) : pour les opérations numériques.
• matplotlib (importé sous le nom plt) : pour les pour graphiques et des figures dans Python.
• sklearn.preprocessing : pour la normalisation des données.
• sklearn.model_selection : pour la division des données en ensembles d'entraînement et de test, ainsi que la validation croisée.
• sklearn.metrics : pour l'évaluation des modèles.
• sklearn.decomposition : pour l'analyse en composantes principales (PCA).
• sklearn.neural_network : pour le modèle de classification MLP.
• sklearn.ensemble : pour le modèle de classification RandomForest.
• sklearn.linear_model : pour le modèle de classification LogisticRegression.
• sklearn.tree : pour le modèle de classification DecisionTree.
• sklearn.neighbors : pour le modèle de classification KNeighbors.
• sklearn.discriminant_analysis : pour le modèle de classification LinearDiscriminantAnalysis.
• sklearn.naive_bayes : pour le modèle de classification GaussianNB.
• sklearn.svm : pour le modèle de classification SVC.
• imblearn.over_sampling : pour l'échantillonnage SMOTE (Synthetic Minority Over-sampling Technique).
• warnings : pour ignorer les avertissements.

3.4.1 Chargement des données

Nous chargeons les données à partir de fichiers CSV. Nous avons utiliser les fichiers suivants :
• "data_36.csv" : contient les données associées à l'attaque DDoS.
• "data_2.csv" : contient les données associées aux autres catégories.

Dans cette étape nous avons procéder à la concaténation des deux fichiers pour avoir un seul ensemble de données.

3.4.2 Sélection des attributs

Tout d'abord on procède à une sélection des attributs utiles pour faire l'apprentissage, ces attributs fournissent des informations sur les caractéristiques des paquets réseau et sont utilisés pour analyser et détecter des comportements malveillants ou suspects dans le trafic réseau.

TABLE 3.6: Les attributs utilisés du dataset Bot-Iot dans notre approche

L'attribut	Les informations fournies
Proto	Représente le protocole réseau utilisé pour la communication. Par exemple, les valeurs peuvent être "tcp" (Transmission Control Protocol), "udp" (User Datagram Protocol), "arp" (Address Resolution Protocol), etc.
stddev	Représente l'écart-type des temps de réponse des paquets réseau. Cela peut donner une indication sur la stabilité ou la variabilité des temps de réponse.
min	Représente la valeur minimale des temps de réponse des paquets réseau.
state	Représente l'état de la connexion réseau. Par exemple, les valeurs peuvent être "RST" (Reset), "REQ" (Request), "FIN" (Finish), "CON" (Established), etc.
mean	Représente la valeur moyenne des temps de réponse des paquets réseau.
drate	Représente le débit de données, c'est-à-dire la quantité de données transmise par unité de temps.
srate	Représente le taux de succès des paquets réseau, c'est-à-dire le pourcentage de paquets réussis par rapport au total des paquets envoyés.
max	Représente la valeur maximale des temps de réponse des paquets réseau.
category	représente la catégorie ou la classe à laquelle appartient l'observation. le contexte du dataset "bot-iot", il peut s'agir de différentes catégories de trafic réseau, comme "Reconnaissance", "DDoS" (Distributed Denial of Service), "Normal", etc.

Ensuite, nous avons procédé à une exploration initiale des données en affichant les deux premières lignes du jeu de données pour obtenir un aperçu rapide. Nous avons également vérifié la forme des données pour connaître le nombre de lignes et de colonnes dans notre jeu de données.

Pour assurer la qualité des données, nous avons vérifié la présence de valeurs manquantes à l'aide de la méthode **"isna().sum()"**. Cela nous a permis de quantifier le nombre de valeurs manquantes dans chaque colonne, ce qui est essentiel pour décider des actions à entreprendre pour le nettoyage des données.

Ensuite, nous avons préparé les caractéristiques en extrayant les colonnes sélectionnées du jeu de données brut et en les stockant dans la variable X. Cependant, certaines des caractéristiques étaient catégoriques, telles que "proto" et "state", et ont nécessité un encodage one-hot pour être utilisées dans les modèles d'apprentissage automatique. Nous avons donc appliqué l'encodage one-hot à l'aide de la méthode **"pd.get_dummies(X)"** et ajouté les nouvelles co-

lonnes correspondant aux différentes catégories à X.

Enfin, nous avons préparé la variable cible en extrayant la colonne "category" du jeu de données brut et en la stockant dans la variable y. Pour mieux représenter notre problème, nous avons appliqué un mapping à la colonne "category" pour la transformer en une variable binaire, où les catégories "Reconnaissance", "DoS", "Theft" et "DDoS" ont été regroupées en une classe positive (1), tandis que la catégorie "Normal" a été assignée à la classe négative (0).

Pour garantir l'impartialité des analyses futures, nous avons également effectué une permutation aléatoire des indices et réorganisé les données X et y en fonction de cet ordre. Ainsi, notre jeu de données est maintenant prêt pour être utilisé dans les prochaines étapes de modélisation et d'évaluation.

3.4.3 Prétraitement des données

Dans cette section de prétraitement des données, nous avons effectué plusieurs étapes pour préparer nos données pour l'entraînement et l'évaluation de notre modèle.

Tout d'abord, nous avons divisé nos données en ensembles de formation et de test à l'aide de la fonction **train_test_split**. Cela nous permet de séparer une partie de nos données pour l'évaluation de notre modèle. Nous avons spécifié un ratio de 70% pour l'ensemble de formation et 30% pour l'ensemble de test. De plus, nous avons utilisé l'option **stratify=y** pour maintenir la répartition des classes dans les deux ensembles.

Ensuite, pour gérer le déséquilibre de classes dans notre ensemble de formation, nous avons appliqué la technique **SMOTE (Synthetic Minority Over-sampling Technique)**. SMOTE génère de nouveaux échantillons synthétiques pour la classe minoritaire afin de les équilibrer avec la classe majoritaire. Dans notre cas, nous avons spécifié une stratégie d'échantillonnage de 0.5, ce qui signifie que nous voulons augmenter le nombre d'échantillons de la classe minoritaire jusqu'à atteindre 70% de la taille de la classe majoritaire.

Après l'application de SMOTE, nous avons de nouveau divisé notre ensemble de formation en ensembles de formation et de test. Cette fois, nous avons utilisé un ratio de 95% pour l'ensemble de formation et 5% pour l'ensemble de test. Cette étape nous permet d'obtenir un ensemble de formation plus restreint sur lequel nous pouvons entraîner notre modèle.

Par la suite, nous avons appliqué deux techniques de mise à l'échelle des caractéristiques : la standardisation et la mise à l'échelle Min-Max. La standardisation, réalisée à l'aide de la classe ***StandardScaler***, transforme les caractéristiques pour qu'elles aient une moyenne de zéro et un écart-type de un. La mise à l'échelle Min-Max, effectuée à l'aide de la classe ***MinMaxScaler***, met à l'échelle les caractéristiques dans une plage spécifiée, généralement entre 0 et 1.

Enfin, nous avons vérifié les dimensions de nos ensembles d'étiquettes *y_train* et *y_test* pour nous assurer que le prétraitement a été effectué correctement.

Ces étapes préliminaires sont essentielles pour préparer nos données avant de les utiliser dans notre modèle d'apprentissage automatique.

3.4.4 Entraînement et évaluation des modèles

Chaque modèle est entraîné séparément sur les données d'entraînement préalablement prétraitées (ici, les données sont mises à l'échelle avec le StandardScaler ou le MinMaxScaler).

Plusieurs modèles de classification sont instanciés, tels que :
KNeighborsClassifier, DecisionTreeClassifier, RandomForestClassifier, SVC, MLPClassifier, LinearDiscriminantAnalysis, GaussianNB et LogisticRegressionLR.

Une boucle est utilisée pour itérer sur chaque modèle. Pour chaque modèle, une validation croisée est effectuée en utilisant StratifiedKFold avec 3 plis.

Les données d'entraînement (X_scal_train et y_train) sont utilisées pour entraîner le modèle.

Différentes métriques de performance, telles que l'AUC et l'exactitude, sont calculées lors de la validation croisée. Les résultats de la validation croisée sont stockés dans la liste "results" pour chaque modèle.

Les performances des différents modèles sont comparées en fonction de l'AUC et de l'exactitude moyenne obtenues lors de la validation croisée. Le modèle avec les meilleures performances est sélectionné pour une utilisation ultérieure.

3.5 Résultats et discussion

Nous avons implémenté un ids utilisant des techniques d'apprentissage automatique et profond. Ces modèles ont été formés et testés sur deux sous-ensembles de données concaténées du Bot-iot Dataset. Plusieurs tests ont été faits afin d'obtenir les bons Hyperparamètres pour chaque modèle. Ces paramètres ne peuvent pas être ajuster durant la phase d'apprentissage, et ils ont un grand impact sur les performances des modèles durant l'apprentissage. Les résultats de test sur les différents modèles sont présentés ci-dessous :

3.5.1 Rapport de classification pour chaque modèle :

Dans cette section nous exposons les résultats de classification binaire de tous les modèles :

Le rapport de classification binaire avec SVM :

Table 3.7: Le rapport de classification binaire (SVM)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.46	0.99	0.63	1482
Accuracy			1.00	599991
Mcro avg	0.73	0.99	0.81	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme SVM dans les deux classes "0" et "1" :

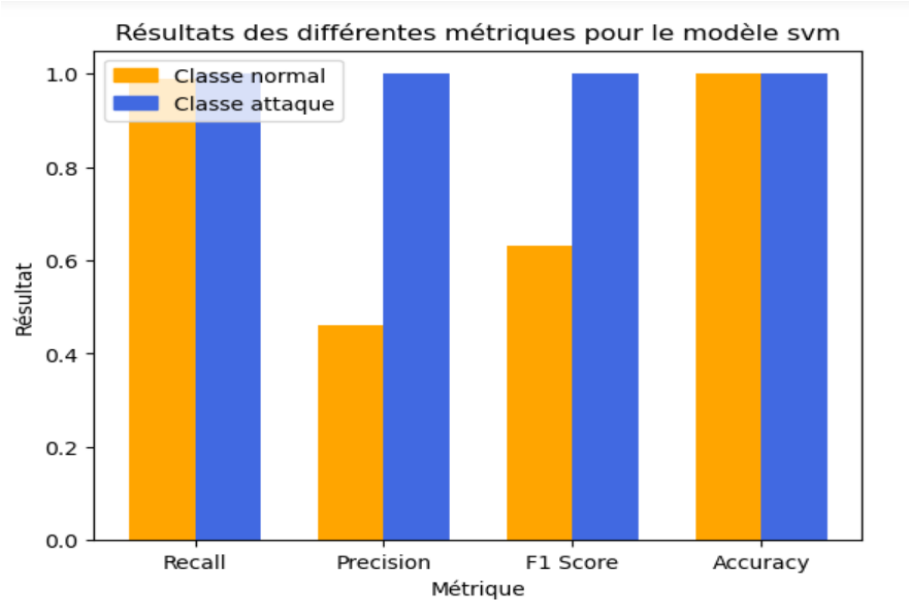

FIGURE 3.6 – Résultats de classification avec SVM

Le rapport de classification binaire avec KNN :

TABLE 3.8: Le rapport de classification binaire (KNN)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.69	1.00	0.81	1482
Accuracy			1.00	599991
Mcro avg	0.84	1.00	0.91	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme KNN dans les deux classes "0" et "1" :

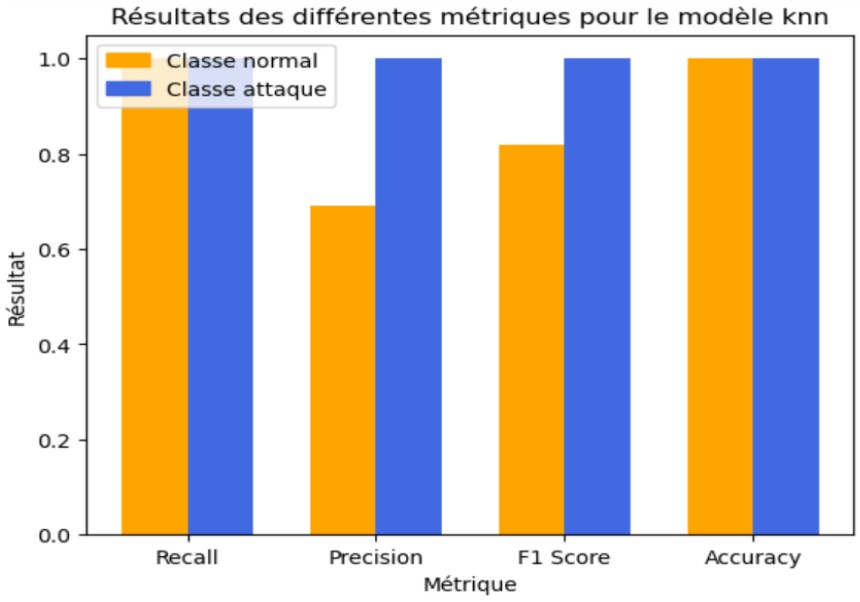

FIGURE 3.7 – Résultats de classification avec KNN

Le rapport de classification binaire avec DT :

TABLE 3.9: Le rapport de classification binaire (DT)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.47	1.00	0.64	1482
Accuracy			1.00	599991
Mcro avg	0.74	1.00	0.82	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme DT dans les deux classes "0" et "1" :

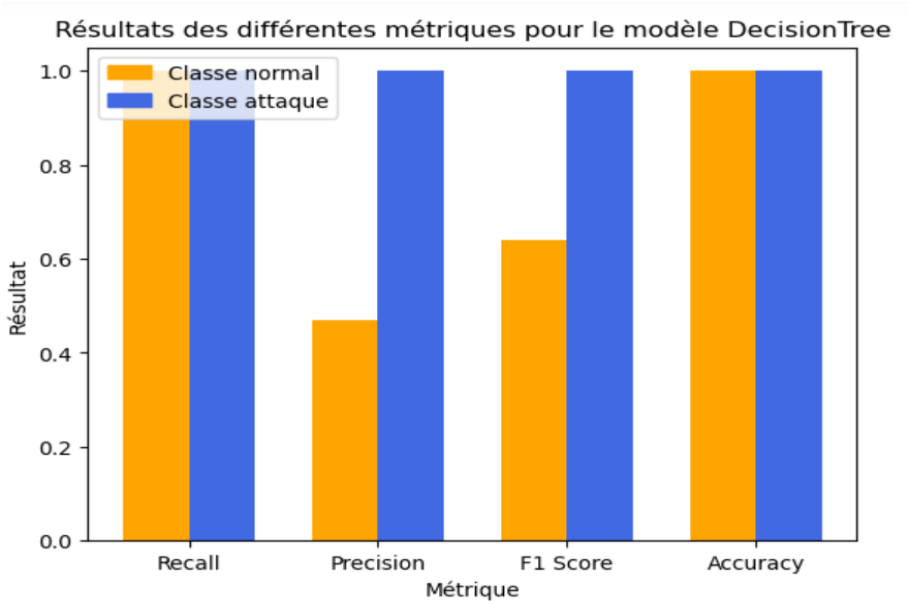

FIGURE 3.8 – Résultats de classification avec DT

Le rapport de classification binaire avec RFC :

TABLE 3.10: Le rapport de classification binaire (RFC)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.74	1.00	0.85	1482
Accuracy			1.00	599991
Mcro avg	0.87	1.00	0.92	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme RFC dans les deux classes "0" et "1" :

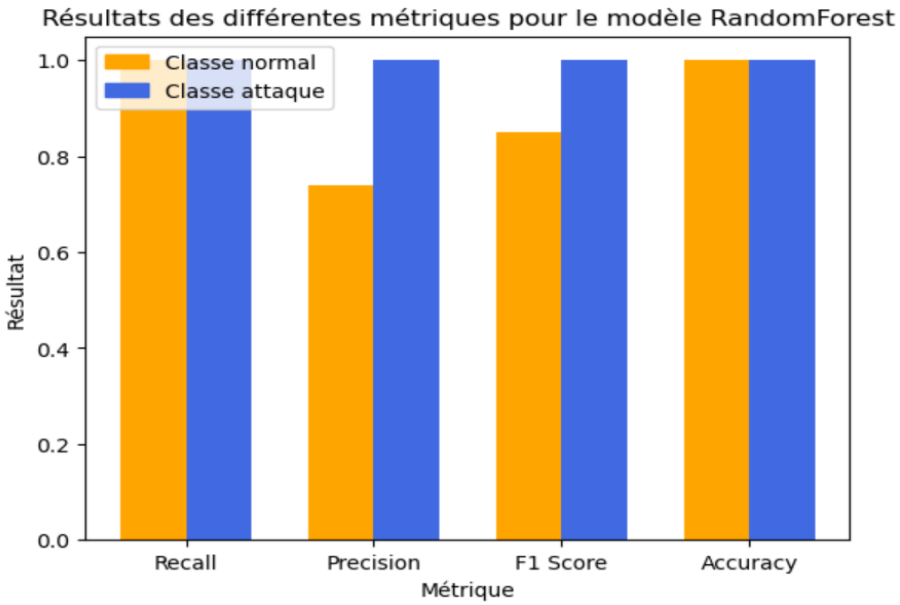

FIGURE 3.9 – Résultats de classification avec RFC

Le rapport de classification binaire avec LR :

TABLE 3.11: Le rapport de classification binaire (LR)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.41	0.99	0.57	1482
Accuracy			1.00	599991
Mcro avg	0.70	0.99	0.78	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme LR dans les deux classes "0" et "1" :

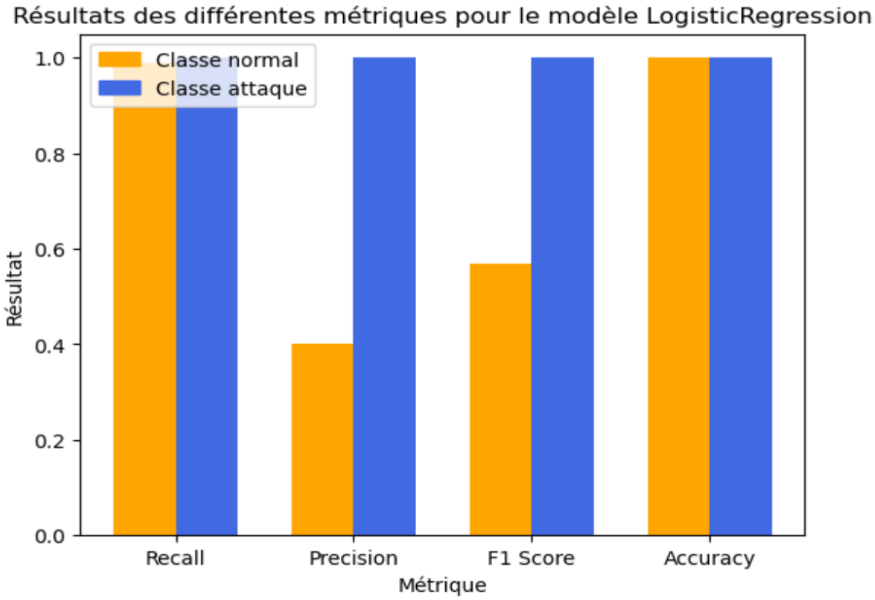

FIGURE 3.10 – Résultats de classification avec LR

Le rapport de classification binaire avec LDA :

TABLE 3.12: Le rapport de classification binaire (LDA)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.42	0.99	0.59	1482
Accuracy			1.00	599991
Mcro avg	0.71	0.99	0.79	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme LDA dans les deux classes "0" et "1" :

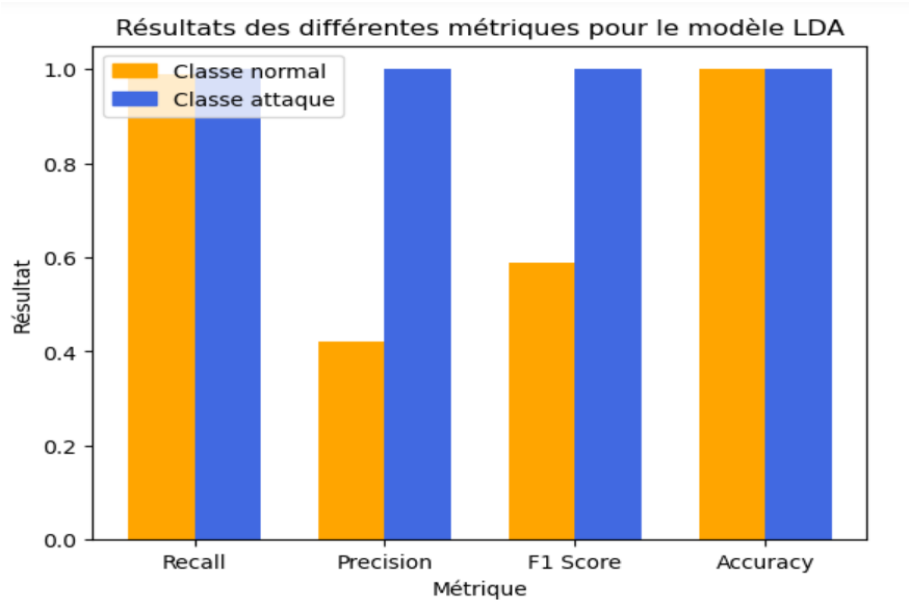

FIGURE 3.11 – Résultats de classification avec LDA

Le rapport de classification binaire avec GNB :

TABLE 3.13: Le rapport de classification binaire (GNB)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.46	0.99	0.62	1482
Accuracy			1.00	599991
Mcro avg	0.73	0.99	0.81	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme GNB dans les deux classes "0" et "1" :

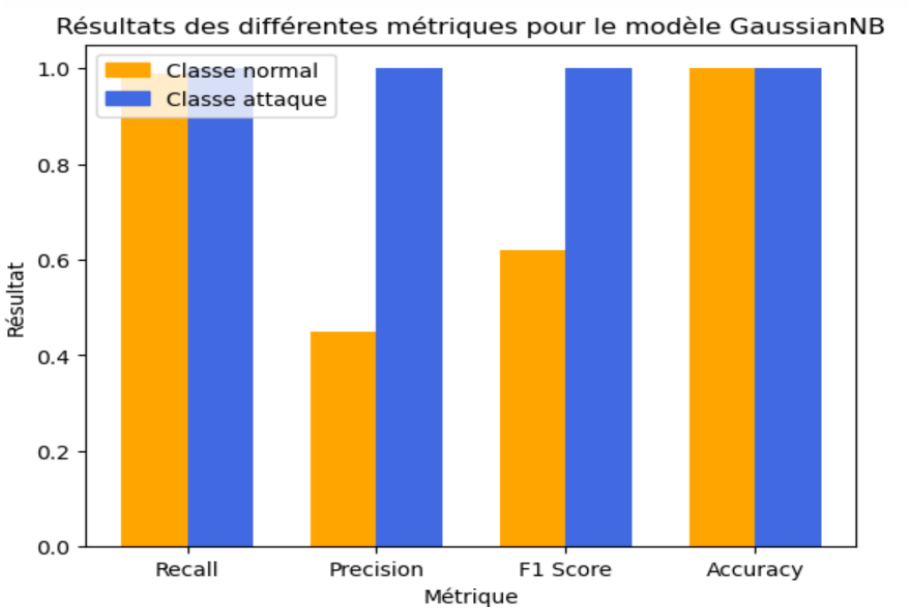

FIGURE 3.12 – Résultats de classification avec GNB

Le rapport de classification binaire avec MLP :

TABLE 3.14: Le rapport de classification binaire (MLP)

	Precision	Recall	F1-score	Support
1 - Attaque	1.00	1.00	1.00	598509
0 - Normal	0.40	0.99	0.57	1482
Accuracy			1.00	599991
Mcro avg	0.70	0.99	0.78	599991
Weighted avg	1.00	1.00	1.00	599991

Le graphe ci-dessous montre la variation des métriques de performances de l'algorithme MLP dans les deux classes "0" et "1" :

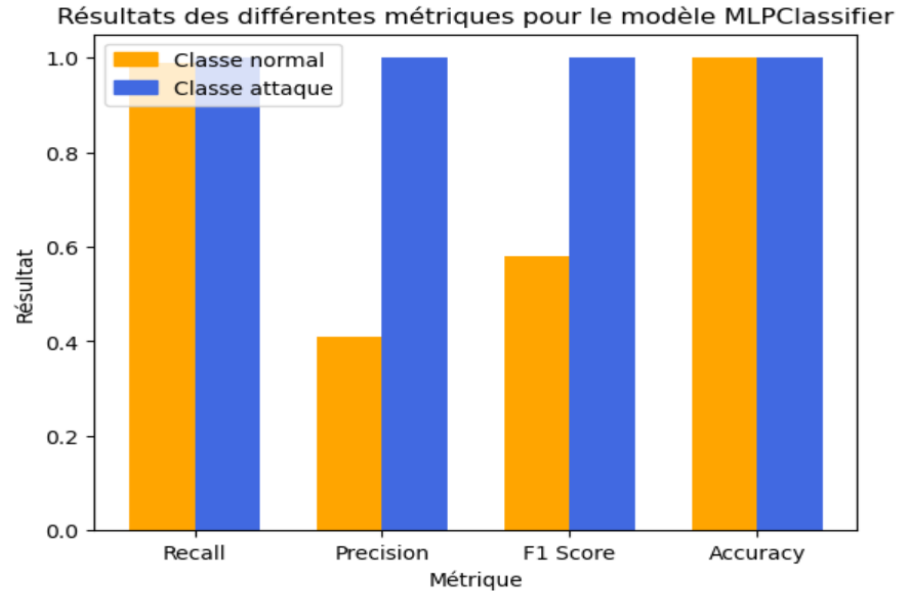

FIGURE 3.13 – Résultats de classification avec MLP

3.5.2 Performances des modèles pour la détection de botnet

Cette sous-section contient l'ensemble des métriques de performances de tous les algorithmes sur le jeu de données Bot-iot selon les métriques suivantes :

TABLE 3.15: Évaluation des performances pour la détection de botnet dans le jeu de données Bot-iot

	SVM	KNN	DT	RFC	LR	LDA	GNB	MLP
Accuracy	0.997	0.998	0.997	0.999	0.996	0.996	0.997	0.996
Recall	0.997	0.998	0.997	0.999	0.996	0.996	0.997	0.996
F1-score	0.998	0.998	0.998	0.999	0.998	0.998	0.998	0.998
Precision	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999
AUC	0.993	0.997	0.995	0.997	0.992	0.991	0.991	0.993

TABLE 3.16: Matrice de confusion pour les modèles standardiser avec Z-score

		SVM		KNN		DT		RFC	
Confusion matrix	0	1468	14	1476	6	1473	9	1475	7
	1	1744	596765	765	597744	1674	596835	530	597979

TABLE 3.17: Matrice de confusion pour les modèles normaliser avec Min_Max

		LR		LDA		GNB		MLP	
Confusion matrix	0	1466	16	1461	21	1460	22	1469	13
	1	2130	596379	1996	596513	1733	596776	2221	596228

Les figures ci-dessous contiennent la représentation graphique des métriques exposée ci-dessus :

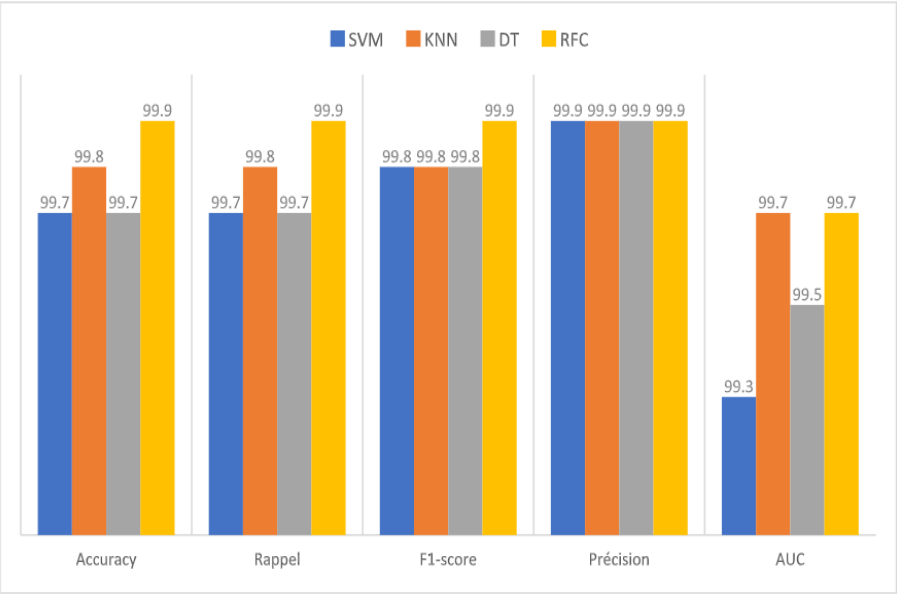

FIGURE 3.14 – Performances des modèles standardisés avec z_score

D'après les résultats illustrés dans la figure 3.12 on déduit que le meilleur modèle avec la standardisation z-score est le RFC avec une Accuracy de 99,9% et un F1-score de 99,9% et 99,7% sur la courbe ROC.

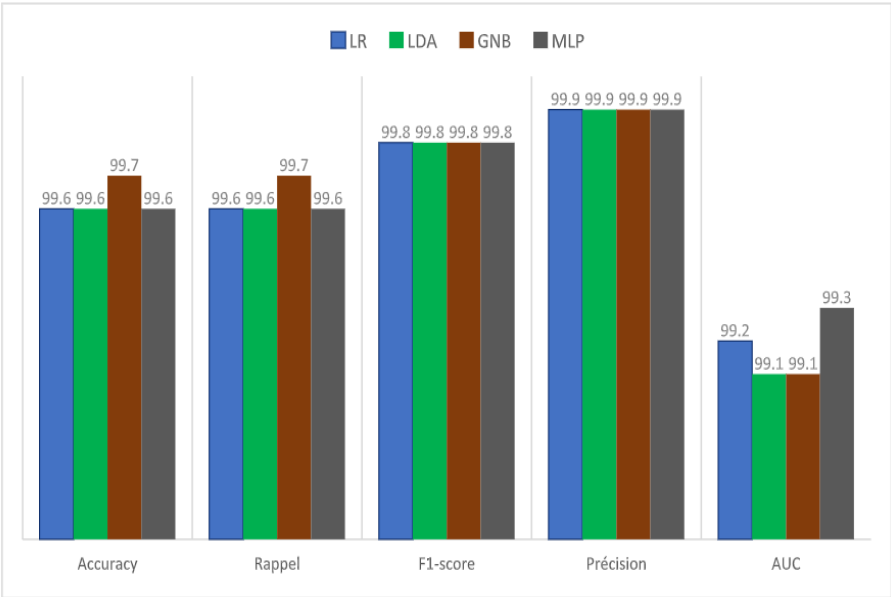

FIGURE 3.15 – Performances des modèles normalisés avec Min_Max

D'après les résultats illustrés dans la figure 3.13 on déduit que le meilleur modèle avec la normalisation Min_Max est le GNB avec une Accuracy de 99,7% et un F1-score de 99,8% et 99,1% sur la courbe ROC.

3.6 Etude comparative

Notre méthode sur l'implémentation d'un IDS en utilisant des techniques de machine et deep learning dans l'iot, offre une approche novatrice pour résoudre le problème d'attaques de Botnet dans l'iot.
Dans cette section nous effectuons une comparaison entre la méthode proposée dans ce mémoire avec une autre méthode alternative.
L'un des principaux critères de comparaison entre les deux méthodes est l'efficacité. Notre méthode se démarque par sa capacité à distingué correctement un trafic malveillant d'un trafic normal. En revanche, la méthode alternative présente une efficacité légèrement inférieure. Un autre critère important est la facilité d'utilisation, notre méthode permet une adoption rapide et une mise en œuvre aisée, tandis que la méthode alternative peut être plus complexe à utiliser.

La figure 3.14 démontre les résultats obtenus en utilisant la méthode alternative qui se porte sur une exploration des ensembles de données existants dans Internet classique, et de catégoriser des caractéristiques sur le jeu de données Botnet-IoT, puis ils ont procédé à l'entraînement de plusieurs modèles d'apprentissage automatique avec chacun des sous-ensembles proposés, puis ils ont évalué les performances de ces modèles.
Et ils ont appliqué la même procédure sur un autre jeu de données de réseaux classiques [43], dans notre cas on s'intéresse seulement au cas appliqué sur le jeu de données de réseau IoT.

		KNN		DT		RFC		SVM		LR		LDA		GNB		MLP	
Accuracy		0.958		0.976		0.964		0.864		0.873		0.870		0.874		0.940	
Recall		0.96		0.98		0.97		0.86		0.86		0.86		0.86		0.92	
F1-score		0.96		0.98		0.97		0.82		0.82		0.82		0.82		0.91	
AUC		0.951		0.973		0.958		0.846		0.695		0.687		0.642		0.922	
Precision	0	0.92		0.96		0.93		0.96		0.92		0.92		0.82		0.96	
	1	0.96		0.98		0.98		0.86		0.86		0.86		0.86		0.92	
Confusion matrix	0	340	72	376	36	364	48	85	327	79	333	81	331	94	318	242	170
	1	31	1952	14	1969	26	1957	4	1979	7	1976	7	1976	20	1963	9	1974

FIGURE 3.16 – Performances des modèles de la méthode alternative [43]

	SVM	KNN	DT	RFC	LR	LDA	GNB	MLP
Accuracy	0.997	0.998	0.997	0.999	0.996	0.996	0.997	0.996
Recall	0.997	0.998	0.997	0.999	0.996	0.996	0.997	0.996
F1-score	0.998	0.998	0.998	0.999	0.998	0.998	0.998	0.998
Precision	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999
AUC	0.993	0.997	0.995	0.997	0.992	0.991	0.991	0.993

FIGURE 3.17 – Performances des modèles de la méthode proposée

Tenant compte des résultats obtenus ci-dessous et en les comparent à nos résultats dans la figure 3.15, nous remarquons une différence dans les métriques de performances, celles de notre

méthode sont plus élevées par rapport à l'autre.

Dans notre cas, nous avons rencontré un problème de déséquilibre de classe qui à conduit à avoir une précision de détection un peut plus faible de la classe "0" (normal) en comparant a celle de la méthode dans [43].

Conclusion

Dans ce chapitre, nous avons présenté les langages, l'environnement et les outils utilisés pour concrétiser notre projet d'implémentation d'un systèmes de détection d'intrusions basés sur l'apprentissage automatique. Le développement et la modification des données étaient un problème auquel nous étions confrontés et qui nous prenait beaucoup de temps à résoudre, et l'un de ces problèmes était que les appareils disponibles pour travailler étaient faibles (RAM ,GPU ,Processeur) et ne remplissaient pas leur objectif.
Grâce à l'utilisation de bibliothèques telles que scikit-learn et pandas, nous avons pu mettre en œuvre différentes étapes essentielles du processus, notamment la préparation des données. Nous avons ensuite entraîné et évalué plusieurs modèles en utilisant un ensemble de données spécifique à l'Internet des objets. Les résultats obtenus sont prometteurs en termes de détection d'intrusions. Les modèles de classification utilisés ont démontré de bonnes performances en termes d'Accuracy, F1-score et AUC, notamment pour le modèle RFC qui s'avère le plus efficace par rapport aux autres.

Conclusion générale

Au cours de ce travail, nous avons examiné l'importance croissante des outils numériques et leur intégration dans tous les aspects de la vie quotidienne. Cependant, cette évolution rapide a également engendré de nouvelles menaces et attaques contre les systèmes informatiques. Afin de relever ces défis, nous avons exploré les possibilités offertes par l'intelligence artificielle, en particulier le machine learning et le deep learning, dans le domaine de la sécurité informatique.

Dans le chapitre 1, nous avons présenté un aperçu de la sécurité informatique, en mettant en évidence les différents types de menaces auxquelles les systèmes sont confrontés et les mécanismes de réponse utilisés pour les contrer. Nous avons également défini le machine learning et exploré les différents types d'algorithmes utilisés, en mettant l'accent sur leur application à la sécurité informatique.

Le chapitre 2 était consacré à l'état de l'art de la détection des attaques basées sur les botnets. Nous avons examiné la littérature existante, analysant les différentes méthodes utilisées, les méthodes de validation, les avantages et les critiques associées à chaque approche. Une analyse comparative nous a permis de mettre en évidence les forces et les faiblesses des différentes propositions.

Notre objectif principal de recherche était de proposer et d'implémenter une approche de détection des botnets dans les réseaux IoT afin de protéger ces réseaux sans perturber leur fonctionnement. Dans le chapitre 3, nous avons présenté notre propre solution pour la détection d'attaques basées sur les botnets en utilisant des approches de machine learning et deep learning. Nous avons décrit le jeu de données utilisé, le protocole de réalisation des modèles, ainsi que l'implémentation et l'évaluation des performances. Une étude comparative a été réalisée pour évaluer l'efficacité de notre solution par rapport aux approches existantes.

En conclusion, ce mémoire a mis en évidence l'importance cruciale du machine learning et du deep learning dans la détection des attaques basées sur les botnets dans l'Internet des objets (IoT). Les résultats obtenus avec notre solution sont raisonnables et satisfaisants en les comparant avec d'autres travaux connexes.

Perspectives

Plusieurs perspectives peuvent être envisagées pour améliorer davantage ce travail. Tout d'abord, il serait intéressant d'élargir notre portée en testant nos modèles de détection sur des ensembles de données plus vastes et plus variés, qui représentent différentes conditions et scénarios d'utilisation de l'IoT. Cela nous permettra d'évaluer les performances de notre approche dans des contextes réels et de valider sa capacité à généraliser au-delà de notre ensemble de données initial.

De plus, une autre perspective prometteuse serait d'explorer l'intégration de techniques de renforcement pour améliorer la capacité de notre système à détecter et à se défendre contre les attaques basées sur les botnets. Les méthodes de renforcement permettent d'apprendre des politiques de décision optimales en interagissant avec l'environnement, ce qui pourrait être appliqué pour renforcer la sécurité de l'IoT face à ces menaces..

Bibliographie

[1] Pradyumna Gokhale, Omkar Bhat et Sagar Bhat. "Introduction to IOT". In : *International Advanced Research Journal in Science, Engineering and Technology* (2018).

[2] Émilie Bout et Valeria Loscri. "Le machine learning, nouvelle porte d'entrée pour les attaquants d'objets connectés". In : *The Conversation* (2021).

[3] Godwin Thomas et Mary-Jane Sule. "A service lens on cybersecurity continuity and management for organizations' subsistence and growth". In : *Organizational Cybersecurity Journal : Practice, Process and People* (2022).

[4] ABDERRAHIM SEBRI. "Cours Sécurité des Systèmes Informatique Support de Cours pour Sécurité des Systèmes d'Informations". In : (2022).

[5] BP Patil, KG Kharade et RK Kamat. "Investigation on data security threats & solutions". In : *International Journal of Innovative Science and Research Technology* (2020).

[6] Kevin Cacciapaglia. "Analyse sur les différentes cyberattaques informatiques". Mémoire de fin de cycle. Haute école de gestion de Genève, 2018.

[7] ANDRIASALAMA Anthonio Hanitra. "DETECTION D'ATTAQUE SUR LES SYSTEMES INFORMATIQUES A BASE D'ALGORITHME DE MACHINE LEARNING". Thèse de doctorat. 2018.

[8] IKRAME NOUAR. "La détection des attaques Botnet dans l'Industrie Internet des objets (IIoT)". In : (2022).

[9] HARROUZ AHMED AMINE AMIRA BOUBAKEUR. "Utilisation des métaheuristiques pour la résolution du problème de sélection d'attributs : Application à la détection d'intrusions". In : (2013).

[10] Aurélien Géron. *Machine Learning avec Scikit-Learn : Mise en oeuvre et cas concrets*. Dunod, 2019.

[11] Hacene BELLAHMER. "Implémentation et évaluation d'un modèle d'apprentissage automatique pour l'estimation de la valeur marchande de propriétés immobilières". In : (2020).

[12] Ludovic DE MATTEIS. "Introduction à l'apprentissage automatique". In : (2022).

[13] Aurélien Vannieuwenhuyze. "Intelligence artificielle vulgarisée". In : *Le Machine Learning et le Deep Learning par la pratique* (2019).

[14] Batta Mahesh. "Machine learning algorithms-a review". In : *International Journal of Science and Research (IJSR).[Internet]* (2020).

[15] Hadj-Tayeb Karima. "Approche de partitionnement pour un apprentissage non supervisé des Usagers du Web (Amélioration de l'approche k-means)". In : (2021).

[16] Alberto Bietti. *Latent dirichlet allocation*. Rapp. tech. mai 2012, working paper, http ://alberto. bietti. me/files/rapport-lda. pdf, 2012.

[17] Stiphen Chowdhury, Na Helian et Renato Cordeiro de Amorim. “Feature weighting in DBSCAN using reverse nearest neighbours”. In : *Pattern Recognition* (2023).

[18] Farzana Anowar, Samira Sadaoui et Bassant Selim. “Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne)”. In : *Computer Science Review* (2021).

[19] Dietmar PF Möller. “Machine Learning and Deep Learning”. In : *Guide to Cybersecurity in Digital Transformation : Trends, Methods, Technologies, Applications and Best Practices*. Springer, 2023, p. 347-384.

[20] Claude Touzet. *les réseaux de neurones artificiels, introduction au connexionnisme*. 1992.

[21] Farhad Mortezapour Shiri et al. “A Comprehensive Overview and Comparative Analysis on Deep Learning Models : CNN, RNN, LSTM, GRU”. In : *arXiv preprint arXiv :2305.17473* (2023).

[22] Yassine HADDAB. “Introduction à l’internet des objets (IdO – IoT)”. In : (2022).

[23] Ahmed Nait-Sidi-Moh, David Durand, Jérôme Fortin et al. “Internet des objets et interopérabilité des flux logistiques : état de l’art et perspectives”. In : *Université de Picardie Jule Verne* (2015).

[24] Lerina Aversano et al. “A systematic review on Deep Learning approaches for IoT security”. In : *Computer Science Review* (2021).

[25] Satish Pokhrel, Robert Abbas et Bhulok Aryal. “IoT security : botnet detection in IoT using machine learning”. In : *arXiv preprint arXiv :2104.02231* (2021).

[26] Arvind Prasad et Shalini Chandra. “VMFCVD : an optimized framework to combat volumetric DDoS attacks using machine learning”. In : *Arabian Journal for Science and Engineering* (2022).

[27] Santha devi D. “IoT Malware Detection using Machine Learning Ensemble Algorithms”. In : *International Journal of Advance Science and Technology* (2020).

[28] Dr Nookala Venu, AArun Kumar et Mr A Sanyasi Rao. “BOTNET ATTACKS DETECTION IN INTERNET OF THINGS USING MACHINE LEARNING”. In : *NEUROQUANTOLOGY* (2022).

[29] Abhishek Raghuvanshi et al. “Intrusion detection using machine learning for risk mitigation in IoT-enabled smart irrigation in smart farming”. In : *Journal of Food Quality* (2022).

[30] Guilherme de O Kfouri et al. “Design of a Distributed HIDS for IoT Backbone Components.” In : 2019.

[31] Vitor Hugo Bezerra et al. “One-class classification to detect botnets in IoT devices”. In : 2018.

[32] Shreehar Joshi et Eman Abdelfattah. “Efficiency of different machine learning algorithms on the multivariate classification of IoT botnet attacks”. In : IEEE. 2020.

[33] Miloud Bagaa et al. “A machine learning security framework for iot systems”. In : *IEEE Access* (2020).

[34] Fereshteh Abbasi, Marjan Naderan et Seyyed Enayatallah Alavi. “Intrusion detection in iot with logistic regression and artificial neural network : further investigations on n-baiot dataset devices”. In : *Journal of Computing and Security* (2021).

[35] Owais Bukhari et al. “Anomaly detection using ensemble techniques for boosting the security of intrusion detection system”. In : *Procedia Computer Science* (2023).

[36] Abdellatif El Ghazi et Ait Moulay Rachid. “Machine learning and datamining methods for hybrid IoT intrusion detection”. In : 2020.

[37] M Tech Scholar Rakhi Sarathe et Sumit Sharma. “IOT Network Malicious Session Detection by KNN and MothFlame Optimization Algorithm”. In : *International Journal of Scientific Research Engineering Trends* (2022).

[38] Hamouda Djallel. “Un système de détection d’intrusion pour la cybersécurité”. Mémoire de fin de cycle. Université de 8 Mai 1945 – Guelma -, 2020.

[39] Thabet ilyas. “Recalage des images médicales par apprentissage profond”. Memoire de fin de cycle. Université Larbi Tebessi, 2021.

[40] Mamoun Alazab et al. “Deep learning for cyber security applications : A comprehensive survey”. In : *TechRxiv* (2021).

[41] Chang-Le Zhong, Zhen Zhu et Ren-Gen Huang. “Study on the IOT architecture and gateway technology”. In : 2015.

[42] Ming Zhong, Yajin Zhou et Gang Chen. “Sequential model based intrusion detection system for IoT servers using deep learning methods”. In : *Sensors* (2021).

[43] Nadim Elsakaan et Kamal Amroun. “A Comparative Study of Machine Learning Binary Classification Methods for Botnet Detection”. In : 2021.

Résumé

L'Internet des objets (IoT) représente l'une des révolutions numériques les plus importantes, offrant de nombreuses opportunités en interconnectant tous les objets du quotidien. Cependant, en raison de leur nature spécifique, ces objets sont plus vulnérables aux attaques des hackers. Les botnets constituent l'une des plus grandes menaces actuelles pour Internet, grâce à leur capacité à mener des attaques distribuées par déni de service (DDoS) à tout moment. Ces attaques deviennent encore plus dévastatrices lorsqu'elles exploitent les capacités de calcul des objets connectés, qui se transforment ainsi en unités de traitement phénoménales lorsqu'elles sont combinées. Afin de faire face à cette problématique, de nombreux travaux ont été réalisés pour anticiper la propagation des botnets dans le contexte de développement de l'IoT, et pour proposer des solutions basées sur l'apprentissage automatique et l'apprentissage profond.

Notre modèle proposé aborde la problématique de sécurité liée aux menaces posées par les bots. Différents algorithmes d'apprentissage automatique tels que K plus proches voisins (KNN), la machine à vecteur de support (SVM), l'arbre de décision (DT), les forets aléatoire (RFC), le modèle de Bayes naïf, la régression logistique (LR), le modèle LDA et le réseau neuronal artificiel à perceptron multicouche (MLP ANN) ont été utilisés pour développer un modèle où les données sont entraînées à l'aide de l'ensemble de données BoT-IoT. Le meilleur algorithme a été sélectionné en se basant sur un point de référence établi en fonction du pourcentage de l'accuracy et de la surface sous la courbe des caractéristiques de fonctionnement du récepteur (ROC AUC). La technique de suréchantillonnage synthétique des minorités (SMOTE) a été combinées avec les algorithmes d'apprentissage automatique (MLA). Enfin nous avons procédé à une évaluation des performances de ces modèles.

Mots clés : IoT, attaque, botnet, apprentissage profond, apprentissage automatique, DDoS, SMOTE.

Abstract

The Internet of Things (IoT) represents one of the most significant digital revolutions, offering numerous opportunities by interconnecting everyday objects. However, due to their specific nature, these objects are more vulnerable to hacker attacks. Botnets are one of the biggest current threats to the Internet, thanks to their ability to launch distributed denial-of-service (DDoS) attacks at any time. These attacks become even more devastating when they exploit the computing capabilities of connected objects, transforming them into powerful processing units when combined.

To address this issue, extensive research has been conducted to anticipate the spread of botnets in the context of IoT development and propose solutions based on machine learning and deep learning.

Our proposed model addresses the security issue posed by bot threats. Various machine learning algorithms, such as K-nearest neighbors (KNN), support vector machines (SVM), decision trees (DT), random forests (RFC), naive Bayes, logistic regression (LR), linear discriminant analysis (LDA), and multi-layer perceptron artificial neural network (MLP ANN), have been utilized to develop a model trained on the BoT-IoT dataset. The best algorithm was selected based on a reference point established using accuracy percentage and the area under the receiver operating characteristic curve (ROC AUC). Synthetic Minority Over-sampling Technique (SMOTE) was combined with the machine learning algorithms (MLAs). Finally, we evaluated the performance of these models.

Keywords : IoT, attack, botnet, deep learning, machine learning, DDoS, SMOTE.