République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A. Mira de Béjaïa
Faculté des Sciences Exactes
Département d'Informatique

Mémoire Fin de cycle master

En vue de l'obtention du diplôme de master recherche en informatique

Option : Intelligence Artificiel

Thème

Apprentissage automatique pour la détection d'intrusion dans un système informatique

Réalisé par :

* Tighzer Yanis
* Mouzaia Raouf

Devant le jury composé de

• **Examinateur 1** : El BOUHISSI .H	M.A.B	Université de Béjaïa
• **Examinateur** 2 : MEHAOUED.K	M.C.A	Université de Béjaïa
• **Encadrant** : AMROUN .K	Professeur	Université de Béjaïa

Année universitaire : 2020-2021

Remerciement

En premier lieu, nous remercions Dieu le très haut qui nous a donné le courage et la volonté de réaliser ce modeste travail.

Nous tenons à saisir cette occasion et adresser nos profonds remerciements et nos profondes reconnaissances à toutes personnes qui nous ont aidés de près ou de loin dans la réalisation de ce mémoire.

Nous remercions M. Amroun Kamal en tant que Directeur de mémoire, pour ses précieux conseils et son orientation tout au long de notre recherche.

Nos remerciements aux membres de jury qui ont accepté de juger notre travail.

Enfin nous exprimons notre profonde reconnaissance à tous responsables et enseignants de l'université de Bejaia qui ont contribué à notre formation.

Dédicace

Je dédie ce modeste travail

a tous ceux qui me connaissent, en

particulier,a la mémoire de mon

père et mon frère ainsi à ma mère.

A mes frères et sœur.
A tous mes amis et collègues.

Sans oublier tous les professeurs qui

ont contribuéà ma formation de

l'enseignement primaire, moyenne et

secondaire jusqu'à l'enseignement supérieur.

Yanis.

Table des matières

INTRODUCTION GENERAL

CHAPITRE 1 *Sécurité informatique et systèmes de détection d'intrusions*

CHAPITRE 2 *Machine Learning*

CHAPITRE 3 *Etat de l'art sur les IDS*

Liste des figures:

Liste des tableaus

Liste des abréviations :

Abréviation	**signification**
ACL	Access Control List
AFRL	Laboratoire de recherche de l'armée de l'air
APT	Advenced Persistent Threat
ARFF	Attribute relation file format
CART	classification and régression Trees
CSV	Comma separated values
DARPA	Défense Advanced Research Projects Agency
DOS	Denial of service
DR	Detection rat
DT	Decision Tree
FN	Fals negativ
FP	Fals positiv
HIDS	Host Intrusion Détection System
IA	l'intelligence artificielle
IDS	Intrusion detection system
IP	Internet protocol
KNN	K-Notest Neighbors
MLP	Multis layer perceptron
MVA	Mathématiques, Vision, Apprentissage

NIDS	Network Intrusion Détection System
NN	Neuronal network
NSL-KDD	Network Security Layer-Knowledge Discovery in Databases
RFE	Recursive Feature Elimination
RELU	Multi-layer Perceptron
RMSE	Root Mean Squad Error
R2L	Remote To Local
SVM	support a vecteurs machine
TCP	Transmission control protocol
TN	Tru Negativ
TP	True Positiv
U2R	User to Root

Introduction général:

L'utilisation d'Internet et des réseaux informatiques se développe, bien que ces technologies apportent de nombreux avantages aux utilisateurs et à la société en général, elles présentent également plusieurs risques. Les pirates informatiques peuvent exploiter les vulnérabilités, ce qui entraîne des vols, des sabotages et affecte la vie des gens dans le monde entier, il est important de faire évoluer les systèmes de protection.

Le système de détection d'intrusion (IDS) est l'un des outils les plus importants pour prévenir les intrusions dans les systèmes. Toute activité visant à porter atteinte à la confidentialité, l'intégrité, l'accessibilité, ou de tenter de contourner les mécanismes de sécurité d'un réseau informatique est appelée intrusion. En revanche, la détection d'intrusion est le processus de surveiller les incidents dans un système ou un réseau et de les analyser pour trouver les signes d'une intrusion. Tout système logiciel ou matériel ou combinaison des deux, qui est responsable de la détecter les intrusions, est connu sous le nom de système de détection d'intrusion ce mémoire est organisé de la manière suivante.

Le premier chapitre seras consacré à présenter une architecture globale d'un IDS, la définition et le mode de fonctionnement de ce dernier. Ainsi que la classification des IDS et enfin la méthode de détection d'une intrusion.

Le deuxième chapitre seras une présentation et explication global du machine Learning ainsi que les algorithmes que nous allons utiliser dans notre projet.

Dans le troisième chapitre nous présentons quelques travaux sur la détection d'intrusion et le Machine Learning.

Et enfin le quatrième chapitre nous passons a la conception et la mise on œuvre de notre IDS ainsi a seras consacrer a la partie résultat et perspective.

L'annexe contient un complément : Outils et environnement de réalisation et Bibliothèques essentielles pour l'apprentissage automatique en Python.

Chapitre 1 :

Sécurité informatique et systèmes de détection d'intrusions

1 Introduction :

Un IDS est une sonde placée sur un réseau ou un système, et qui vas détecte et répondre à des activités douteuses ou anormales en temps réel ou hors-ligne.

Il y a plusieurs types d'IDS telle que les NIDS (Network Intrusion Détection System) qui se basent sur des analyses réseaux, les HIDS (Host Intrusion Détection System), qui assure la sécurité d'un Hôte, les IDS hybrides, qui est un combinent HIDS et NIDS.

2 Composition d'un IDS :

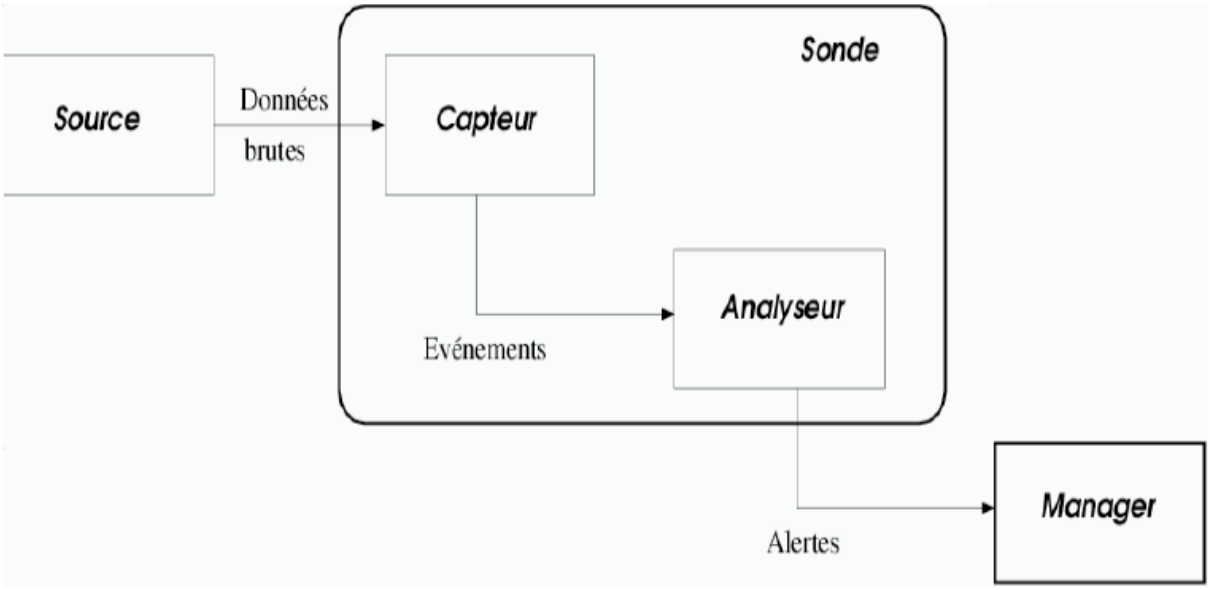


Figure 1 : Composition d'un IDS

Senseur : le senseur est responsable de la collecte d'informations telles que des paquetés d'un réseau ou des données de log.

Analyseur : l'analyseur reçoit l'ensemble des informations des senseur, il est responsable de les analyser et d'identifier si une attaque a lieu ainsi pouvoir agir.

Interface utilisateur : l'interface utilisateur permet aux utilisateurs de visualise leur IDS [1].

3 Caractéristiques des IDS

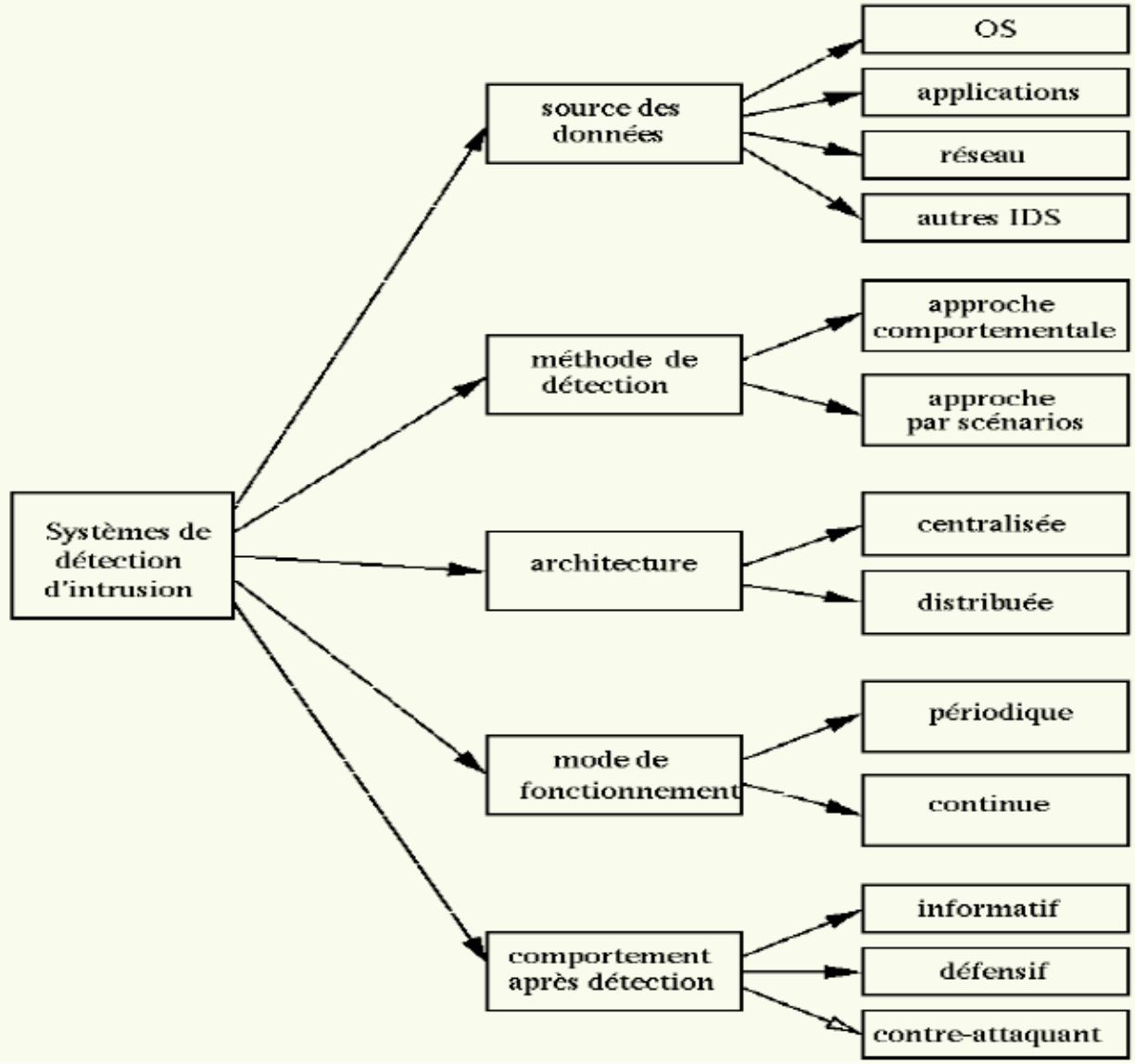
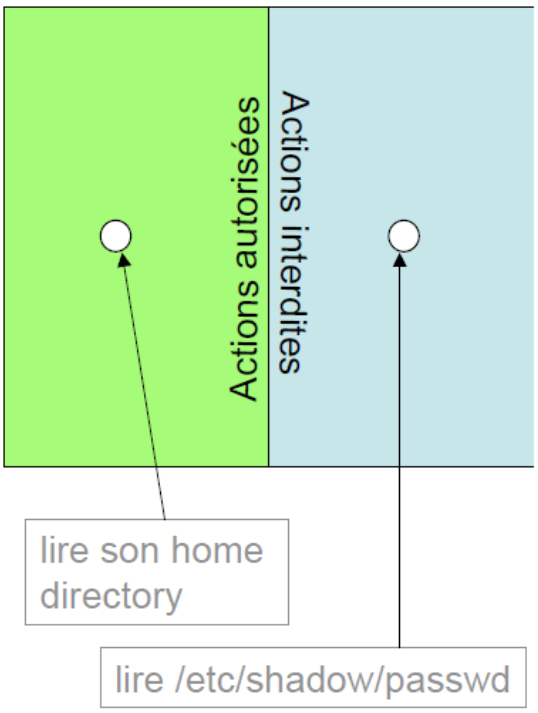


Figure 2 caractéristique d'un IDS

4 Méthode de détection :



Deux possibilités pour détecter les actions abusives :

–Définir et modéliser ce que sont les actions interdites, (scénarios interdits)

–Définir et modéliser ce que sont les actions autorisées, (comportements autorisés) [2].

Une autre différenciation se fait sur la manière de détecter une attaque. Elle peut se baser sur ses approche par scénario ou en se basant sur des approche comportementale. Dans le premier cas, on regarde la suite d'actions effectuées par une personne et on la compare à une attaque connue. Dans le deuxième cas, on regarde le comportement d'une personne et on compare son comportement normal [1].

4.1 Approche comportementale :

Les IDS comportementaux ont pour principale fonction la détection d'anomalie. Leur déploiement nécessite une phase d'apprentissage pendant laquelle l'outil va apprendre le comportement "normal" des flux applicatifs présents sur son réseau [4].

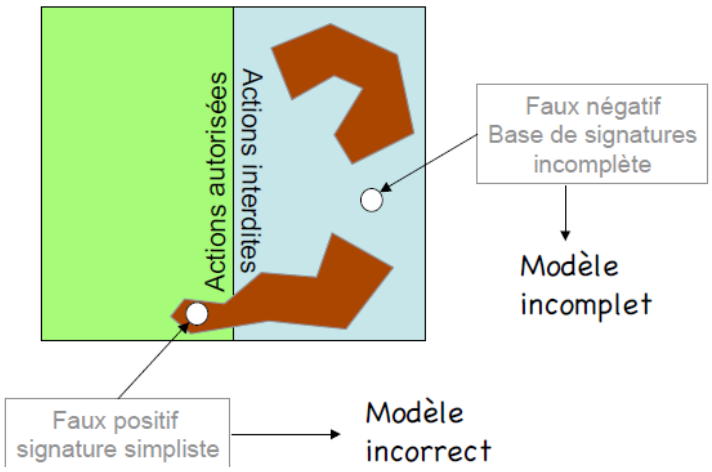


Figure 3 approche comportemental

4.2 Approche par scénario :

Généralement, les IDS réseaux se basent sur un ensemble de signatures qui représentent chacune le profil d'une attaque. Cette approche consiste à rechercher dans l'activité de l'élément surveillé (un flux réseau) les empreintes d'attaques connues, à l'instar des anti-virus [4].

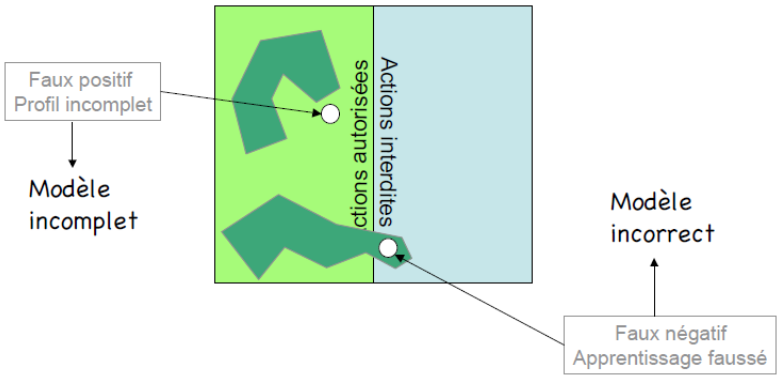


Figure 4 : Approche par scénario

5 Différentes classes d'attaques informatiques :

Une menace est une violation potentielle de la sécurité. Cette menace peut être accidentelle, intentionnelle (attaque), active ou passive, ils existent plusieurs classifications d'attaques informatique selon des critères différents, parmi lesquelles [3] :

5.1 Classification selon l'effet de l'attaque :

Les attaques internes : provenant d'un réseau local ou des utilisateurs.

Les attaques externes : provenant de l'extérieur, fréquemment via internet.

5.2 Classification selon la cible de l'attaque :

Les attaques réseaux: les attaques réseaux s'appuient sur des vulnérabilités liées directement aux protocoles ou à leur implémentation. Il existe un grand nombre. Néanmoins, la plupart d'entre elles ne sont que des variantes des cinq attaques réseaux les plus connues aujourd'hui [3].

Les attaques applicatives: les attaques applicatives s'appuient principalement sur des vulnérabilisées spécifiques aux applications utilisées [3].

Exemples d'attaques:

Ils existent un nombre important de type d'attaque qui menacent les systèmes et les réseaux informatiques, la majorité d'entre elle ne sont que des variantes [3].

Voici les types d'attaques les plus connus :

Attaques de dénis de services : (Denial of service [DOS]) : est une attaque informatique ayant pour but de saturer d'information le système et d'empêcher les utilisateurs de l'utiliser, il peut s'agir de :

L'inondation d'un réseau afin d'empêcher son fonctionnement.

La perturbation des connexions entre deux machines, empêchant l'accès a un service particulier.

L'obstruction d'accès a un service à une personne en particulier [3].

Probing(Sondage) : L'attaquant de cette classe commence par un sondage de la future victime, ce que l'on appelle scan, ce sondage va balayer chaque port IP afin de connaitre les services offerts par le système (OS, topologie du réseau, protections employées,..) une fois se achevé, la machine de l'intrus (celui qui réalise l'intrusion) tente alors d'identifier le système d'exploitation utilisé par cette victime et d'exploiter les informations qu'elle a récolté [3].

Attaques User to Root (U2R) : L'objectif de cette classe d'attaques est d'obtenir la main de l'administrateur système (Root) à partir d'un simple compte utilisateur par l'exploitation des vulnérabilités, Les principales attaques de ce type sont : Eject, Ffbconfig, Fdformat, Load module, Perl, Ps, Xterm [3].

Attaques Remote to User : Dans cette classe d'attaque, l'attaquant essaye d'exploiter les vulnérabilités d'une machine distante afin d'avoir un accès illégal à cette dernière, Pour réussir cette attaque, l'attaquant exploite les bugs des applications installées dans la machine cible [3].

6 Différentes actions que les IDS peuvent mener :

Les principales méthodes utilisées pour signaler et bloquer les intrusions sur les N-IDS sont les suivantes :

Reconfiguration d'équipement tierces (firewall, ACL sur routeurs) : Ordre envoyé par le IDS à un équipement tierce (filtreur de paquets, pare-feu) pour une reconfiguration immédiate dans le but de bloquer un intrus source d'intrusions [3].

Envoi d'une trap SNMP à un hyperviseur tierce : Envoi de l'alerte (et le détail des informations la constituant) sous format d'un datagramme SNMP à une console [3].

Envoi d'un e-mail à un ou plusieurs utilisateurs : Envoi d'un e-mail pour notifier d'une intrusion sérieuse [3].

Journalisation (log) de l'attaque : Sauvegarde des détails de l'alerte dans une base de données centrale [3].

Sauvegarde des paquets suspicieux : Sauvegarde de l'ensemble des paquets réseaux (rawpackets) capturés et/ou seul(s) les paquets qui ont déclenchés une alerte [3].

Démarrage d'une application : Lancement d'un programme extérieur pour exécuter une action spécifique (envoi d'un message sms, émission d'une alerte auditive…) [3].

Envoi d'un "*ResetKill*" : Construction d'un paquet TCP FIN pour forcer la fin d'une connexion[3].

Notification visuelle de l'alerte : Affichage de l'alerte dans une ou plusieurs console(s) de management [3].

7 Conclusion

Dans ce chapitre nous avons abordé différentes notions de la sécurité informatique, on a expliqué les attaques informatiques, leurs classifications et les mécanismes utilisés dans la prévention contre ces attaques, parmi ces mécanisme on a détaillé les systèmes de détection des intrusions vu que c'est notre objectif dans ce mémoire, qui jouent un rôle complémentaire aux mécanismes de sécurité traditionnels.

Nous avons présenté aussi le principe de fonctionnement des IDS ainsi leurs classifications selon différents critères avec leurs avantages et inconvénients, parmi les critères de classification abordés la méthode de détection qui divise les IDS en deux types, les IDS comportementaux et à base de signatures, le principe de ce dernier consiste à rechercher dans l'activité de l'élément surveillé les empreintes (ou signatures) d'attaques connues d'une façon similaire à celle des antivirus, ce type d'IDS est simple d'implémenter mais il présente des difficultés comme la nécessité de mettre à jour d'une façon quotidienne la base de signature, en plus seuls les attaques contenues dans la base sont détectables, cela peut baissé l'efficacité des systèmes de détection des intrusions à base des signature surtout avec l'énorme développement des attaques réseaux d'aujourd'hui, et pour que la détection d'intrusions soit fiable nous avons basé dans notre projet(Détection d'intrusions à base de réseaux de neurones et algorithmes arbre de decision) sur l'approche comportementale qui rend la détection des intrusions inconnues possible et vise à classifier les enregistrements réseau en deux catégories(normal/attaque) en s'appuyant sur les réseaux de neurones et les algorithmes arbre de décision . d'abord c'est quoi machine Learning ?, c'est se que en vas expliquer dans le prochain chapitre: chapitre 2(machine learning) .

Chapitre 2 :

Machine Learning

1 Introduction

Machine Learning, apprentissage automatique ou bien apprentissage machine est une forme de l'intelligence artificielle (IA) qui permet a un système d'apprendre a partir d'un grande jeu de données, il est déjà appliquer depuis des décennies comme les années quatre vingt dix dans les filtres anti spam, ou encore au début des années 2000 dans les jeux vidéos, aujourd'hui machine Learning existe partout ; dans les voiture autonome, la reconnaissance de parole la détection de visage etc. Le monde est entrain de connaitre un nouvel air grâce au machine Learning.

Définition 1.1: L'intelligence artificielle est définie par Marvin Minsky, qui fut un pionnier de l'IA, comme : « la construction de programmes informatiques qui s'adonnent à des tâches qui sont pour l'instant, accomplies de façon plus satisfaisante par des êtres humains car elles demandent des processus mentaux de haut niveau tels que : l'apprentissage perceptuel, l'organisation de la mémoire et le raisonnement critique» [5].

Définition 1.2: Le machine Learning est quant à lui défini en 1959 par Arthur Samuel comme suit: « L'apprentissage automatique est la discipline donnant aux ordinateurs la capacité d'apprendre sans qu'ils soient explicitement programmés.» [6].

2 Types de system d'apprentissage automatique:

Selon la nature du problème traité et le type ainsi le volume de données On distingue quatre grandes catégories: l'apprentissage supervisé, l'apprentissage non supervisé, l'apprentissage semi-supervisé et l'apprentissage avec renforcement.

2.1 Apprentissage supervisé :

Est une méthode ou on utilise des donné étiqueté : des donnée d'entraînement qu'on fournit à l'algorithme comportent les solutions désirées ; et qui est pour but de minimisé l'erreur en comparant sa sortie réelle avec les sorties enseignées

Soit D un ensemble de données, X l'ensemble de caractéristique qui décris ces donnes, un algorithme d'apprentissage supervisé vas trouver une fonction de mapping entre les entrées X et les sorties Y la fonction de mapping décrivant une relation entre X et Y s'appelle un model de prédiction.

Un exemple consiste à prédire le prix d'une voiture à partir des valeurs d'un certain nombre d'attributs ou variables qu'on appelle caractéristiques d'une observation ou « features » en anglais. Ces variables peuvent être le kilométrage, l'âge, la marque, etc. Ils sont également appelés variables explicatives ou prédictives. L'entrainement se fait alors à partir de ces variables et des étiquettes. La catégorie de la variable prédite Y fait décliner l'apprentissage supervisé en deux sous catégories : la classification et la régression, ces deux concepts seront abordés plus tard.

2.2 Apprentissage non supervisé :

L'apprentissage non supervisé consiste en la conception d'un modèle structurant l'information, c'est-à-dire les données utiliser ne sont pas étiquetées, cette méthode permet a l'algorithme de trouver des relations entre les données d'entrer . L'apprentissage non supervisé peut être utilisé pour la réduction de dimension ou l'extraction de variable. Cette tâche consiste à simplifier les données sans perdre trop d'informations ce type d'algorithme est souvent utilisé pour des problèmes de partitionnement le nombre et la nature des partitions ne sont pas connu a priori [6][7].

2.3 Apprentissage semi-supervisé :

L'apprentissage semi-supervisé vise à résoudre les problèmes avec des données non étiquetées à l'aide de l'ensemble d'informations étiquetées.

Un exemple illustrant l'utilisation d'un apprentissage semi-supervisé est le service d'hébergement d'image : Google Photos. Une fois avoir téléchargé des photos de famille sur ce service, le système arrive à reconnaitre qu'une personne A apparaît sur telle ou telle photos et qu'une personne B sur telle autres. Cela est dû à la partie non supervisé de l'algorithme. Une fois que vous aviez identifié ces personnes, juste une étiquette par personne, le système sera capable de nommer les personnes figurant sur chaque photo, ce qui est utile pour des recherches ultérieures [6].

2.4 Apprentissage par renforcement :

Cette méthode d'apprentissage consiste à mettre en place un système à travers lequel l'algorithme va apprendre, par expériences successives, à résoudre un problème en adoptant un comportement idéal. Contrairement aux autres méthodes d'apprentissage il ne dispose pas de donnée mais plutôt il sera confronté a des variables de son environnent dont il utilisera pour concevoir une stratégie de résolution.

Comme le note le "Data Analytics Post", publication spécialisée portée par le master MVA (Mathématiques, Vision, Apprentissage) de l'Ecole normale supérieure Paris-Saclay, "l'apprentissage par renforcement diffère fondamentalement des problèmes supervisés et non supervisés par ce côté interactif et itératif : l'agent essaie plusieurs solutions (on parle 'd'exploration'), observe la réaction de l'environnement et adapte son comportement pour trouver la meilleure stratégie (il 'exploite' le résultat de ses explorations). Un des concepts clés de ce type de problèmes est l'équilibre entre ces phases d'exploration et d'exploitation" [7][8].

3 Approximation de fonction, classification et régression :

3.1 Approximation de fonction

Le terme « modélisation prédictive » désigne un ensemble de méthodes qui permettent d'analyser et d'interpréter des données définies afin d'effectuer une prédiction sur de futurs données. La modélisation prédictive peut être décrite comme un problème mathématique consistant à approximer une fonction de mappage f entre les variables prédictives en entrée X et la variable à prédire Y. C'est ce qu'on appelle un problème d'approximation de fonction. En règle générale, toutes les tâches d'approximation de fonctions peuvent être divisées en tâches de classification et en tâches de régression [9].

3.2 Classification

La classification est définie comme le processus de reconnaissance, de compréhension et de regroupement d'objets a partir des algorithmes de classification qui sont utilisés lorsque la variable à prédire Y est discrète

les algorithmes de classification utilisés dans l'apprentissage automatique utilisent des données d'entraînement en entrée dans le but de prédire la probabilité que les données qui suivent entrent dans l'une des catégories prédéterminées. L'une des applications les plus courantes de la classification est le filtrage des courriers électroniques en "spam" ou "non-spam", comme le font aujourd'hui les principaux fournisseurs de services de courrier électronique

3.3 Régression

Les algorithmes de régression sont quant à eux utilisés lorsque la variable à prédire Y est continue. Comme le cas de prédire le prix d'une voiture.

Exemple :

Prédire le nombre de clics sur un lien ;

Prédire le nombre d'utilisateurs et utilisatrices d'un service en ligne à un moment donné ;

Prédire le prix d'une action en bourse ;

Prédire l'affinité de liaison entre deux molécules ;

Prédire le rendement d'un plant de maïs.

4 Quelques Algorithmes utilisés en machine Learning :

4.1 K plus proches voisins (K-NN) :

Les K-Plus Proches Voisins est une technique et un algorithme d'apprentissage automatique qui peut être utilisés pour la régression et la classification. Les K-Notest Neighbors examinent les étiquettes d'un nombre choisi de points de données entourant un point de données cible, afin de faire une prédiction sur la classe à laquelle appartient le point de données. KNN est un algorithme conceptuellement simple mais très puissant, et pour ces raisons, il est l'un des algorithmes d'apprentissage automatique le plus populaires.

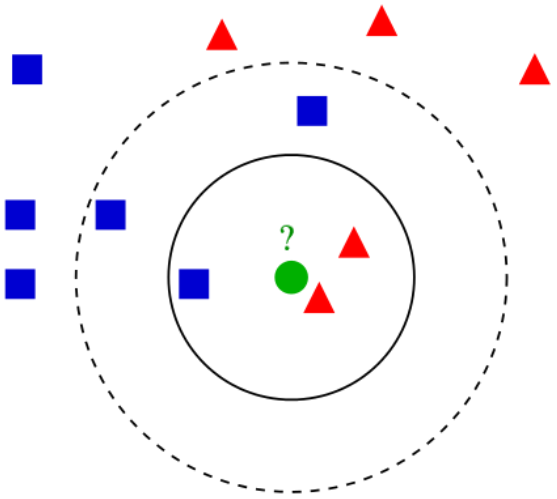


Figure 5 : KNN [10]

KNN examine la distribution des points de données et, en fonction des arguments donnés au modèle, il sépare les points de données en groupes. Une étiquette est ensuite attribuée à ces groupes. L'hypothèse principale d'un modèle KNN est que les points de données/instances qui existent à proximité les uns des autres sont très similaires, tandis que si un point de données est éloigné d'un autre groupe, il est dissemblable de ces points de données.

Un modèle KNN calcule la similarité en utilisant la distance entre deux points sur un graphique. Plus la distance entre les points est grande, moins ils sont similaires. Il existe plusieurs façons de calculer la distance entre des points, mais la mesure de distance la plus courante est la distance euclidienne (la distance entre deux points sur une ligne droite).

KNN est un algorithme d'apprentissage supervisé, ce qui signifie que des étiquettes doivent être attribuées aux exemples de l'ensemble de données et que leurs classes doivent être connues. Il y a deux autres choses importantes à savoir sur KNN. Premièrement, KNN est un algorithme non-paramétrique. Cela signifie qu'aucune hypothèse sur l'ensemble de données n'est faite lorsque le modèle est utilisé. Au contraire, le modèle est entièrement construit à partir des données fournies. Deuxièmement, il n'y a pas de division de l'ensemble de données en ensembles de formation et de test lors de l'utilisation de KNN. KNN ne fait aucune généralisation entre un ensemble de formation et un ensemble de test, donc toutes les données de formation sont également utilisées lorsque le modèle doit faire des prédictions.

Un algorithme KNN passe par trois phases principales au cours de son exécution :

Définir K comme le nombre choisi de voisins.

Calcul de la distance entre un exemple fourni/testé et les exemples de la base de données.

Trier les distances calculées.

Obtenir les étiquettes des K premières entrées.

Retourner une prédiction sur l'exemple de test.

Dans la première étape, K est choisi par l'utilisateur et indique à l'algorithme combien de voisins (combien de points de données environnants) doivent être pris en compte lors de la formulation d'un jugement sur le groupe auquel l'exemple cible appartient. Dans la deuxième étape, notez que le modèle vérifie la distance entre l'exemple cible et chaque exemple de l'ensemble de données. Les distances sont ensuite ajoutées dans une liste et triées. Ensuite, la liste triée est vérifiée et les étiquettes des K premiers éléments sont renvoyées. En d'autres termes, si K est fixé à 5, le modèle vérifie les étiquettes des 5 points de données les plus proches du point de données cible. Lors du rendu d'une prédiction sur le point de données cible, il est important de savoir si la tâche est une tâche de régression ou de classification. Pour une tâche de régression, la moyenne des K premières étiquettes est utilisée, tandis que le mode des K premières étiquettes est utilisé dans le cas de la classification [11].

4.2 Arbre de décision (Decision Tree) :

Les arbres de décision font parti de la catégorie des algorithmes supervisés, elles permettent de prédire une valeur (prédiction) ou une catégorie (classification).
Un arbre de décision est représenté sous forme d'un arbre chaque nœud de l'arbre représente un test (condition) sur une variable, et chacun de ces enfants correspond a une repense possible. Les feuilles correspond a une étiquette

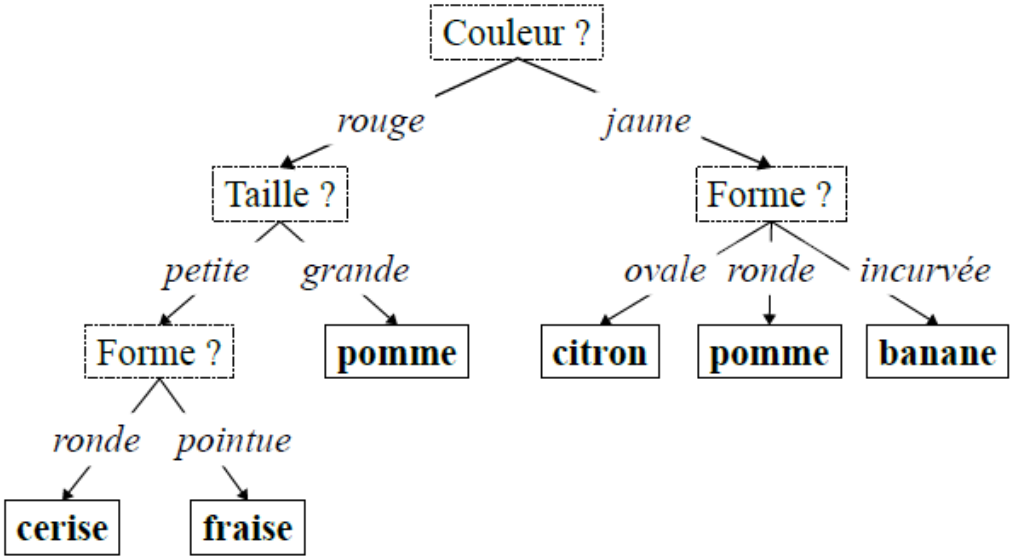


Figure 6 : Exemple d'arbre de décision pour étiqueter un fruit

Il existe plusieurs algorithmes automatiques pour construire les arbres de décision

ID3 développé en 1986 par Ross Quinlan. Il peut être appliquer seulement sur les caractéristique nominales. Il est utilisé pour le classement

C4.5 une extension de ID3 par Ross Quinlan. Il peut être appliquer sur tout les types de caractéristiques. Il est utilisé pour le classement

C5.0 une extension commerciale de C4.5, toujours par Ross Quinlan.

CART (classification and régression Trees): comme C4.5 mais utilise d'autres métrique. Aussi, l'algorithme supporte la régression

Algorithme générale de création d'un arbre de décision :

Déterminer le meilleur caractéristique dans l'ensemble de données d'entrainement.

Diviser les données d'entrainement en sous ensemble contenant les valeurs possibles de la meilleur caractéristique

Générer de manière récursive de nouveaux arbres de décision en utilisant les sous ensembles de donnes créés. Lorsqu'en peut plus classifier les données, on s'arrête.

4.3 SVM (support a vecteurs machine)

Les SVM sont une famille d'algorithmes d'apprentissage automatique qui permet de résoudre des problèmes de classification, régression ou détection d'anomalie.

Les SVM ont été développés dans les années 1990. L'objectif et de trouver un hyperplan dans un espace de N dimension (N est le nombre de caractéristique) qui classifie distinctement les points de données.

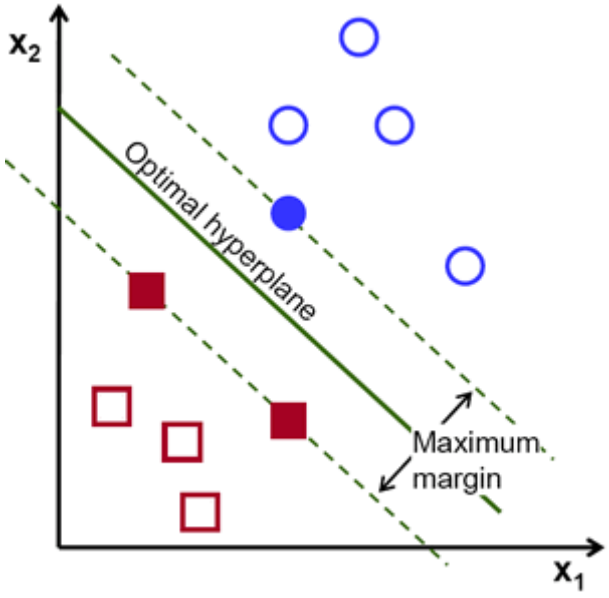


Figure 7 : Hyperplan optimal [12]

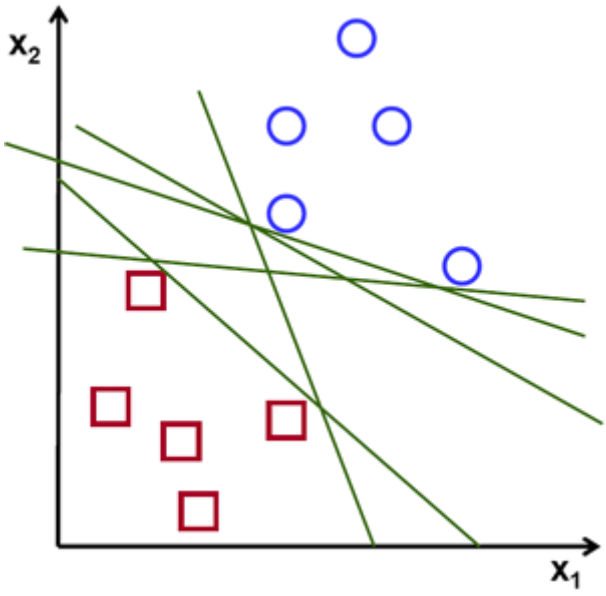


Figure 8 : Hyperplans possibles [12]

Pour séparer les deux classes de points de données, il existe de nombreux hyperplans possibles qui pourraient être choisis. Notre objectif est de trouver un plan qui a la marge maximale, c'est-à-dire la distance maximale entre les points de données des deux classes. La maximisation de la distance de marge fournit un certain renforcement, de sorte que les futurs points de données peuvent être classés avec plus de confiance [12].

4.3.1 Intuition de la grande marge

Dans la régression logistique, nous prenons la sortie de la fonction linéaire et écrasons la valeur dans la plage [0,1] en utilisant la fonction sigmoïde. Si la valeur écrasée est supérieure à une valeur seuil (0,5), nous lui attribuons une étiquette 1, sinon nous lui attribuons une étiquette 0. Dans le SVM, nous prenons la sortie de la fonction linéaire et si cette sortie est supérieure à 1, nous l'identifions avec une classe et si la sortie est -1, nous l'identifions avec une autre classe. Comme les valeurs de seuil sont changées en 1 et -1 dans le SVM, nous obtenons cette plage de valeurs de renforcement([-1,1]) qui agit comme une marge [12].

4.3.2 Fonction de coût et mises à jour du gradient

Dans l'algorithme SVM, nous cherchons à maximiser la marge entre les points de données et l'hyperplan. La fonction de perte qui permet de maximiser la marge est Hing Loss :

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

Le coût est de 0 si la valeur prédite et la valeur réelle sont de même signe. Si elles ne le sont pas, nous calculons alors la valeur de la perte. Nous ajoutons également un paramètre de régularisation à la fonction de coût. L'objectif du paramètre de régularisation est d'équilibrer la maximisation de la marge et la perte. Après avoir ajouté le paramètre de régularisation, les fonctions de coût se présentent comme suit.

$$min_w \lambda \parallel w \parallel^2 + \sum_{i=1}^{n} (1 - y_i \langle x_i, w \rangle)_+$$

Maintenant que nous avons la fonction de perte, nous prenons les dérivées partielles par rapport aux poids pour trouver les gradients. En utilisant les gradients, nous pouvons mettre à jour nos poids.

$$\frac{\delta}{\delta w_k}\lambda \parallel w \parallel^2 \quad = 2\lambda w_k$$

$$\frac{\delta}{\delta w_k}\left(1 - y_i\langle x_i, w\rangle\right)_+ = \begin{cases} 0, & \text{if } y_i\langle x_i, w\rangle \geq 1 \\ -y_i x_{ik}, & \text{else} \end{cases}$$

Lorsqu'il n'y a pas d'erreur de classification, c'est-à-dire que notre modèle prédit correctement la classe de notre point de données, nous devons seulement mettre à jour le gradient du paramètre de régularisation.

$$\boldsymbol{w = w - \alpha \cdot (2\lambda w)}$$

Lorsqu'il y a une mauvaise classification, c'est-à-dire que notre modèle se trompe dans la prédiction de la classe de notre point de données, nous incluons la perte avec le paramètre de régularisation pour effectuer la mise à jour du gradient [12].

$$\boldsymbol{w = w + \alpha \cdot (y_i \cdot x_i - 2\lambda w)}$$

4.4 Réseau de neurone artificiel :

Réseaux de neurone artificiel, l'un des outils fondamental utilisé dans l'apprentissage automatique comme son nom l'indique, il s'agit des Systems inspirer du cerveau humain qui vise a reproduire apprentissage humain.

4.4.1 Concept et définition :

Histoire : L'histoire des réseaux de neurones artificiels revient au1943, où Mac Cullochet Pittsont proposé des neurones formels mimant les neurones biologiques et capables de mémoriser des fonctions booléennes simples. Les réseaux de neurones artificiels réalisés à partir de ce type de neurones sont ainsi inspirés du système nerveux. Ils sont conçus pour reproduire certaines caractéristiques des mémoires biologiques par le fait qu'ils sont massivement parallèles capables d'apprendre et de mémoriser l'information dans les connexions entres neurones, capables de traiter des informations incomplètes

En 1949, Hebb a mis en évidence l'importance du couplage synaptique dans l'apprentissage par renforcement ou dégénérescence des liaisons inter-neuronales lors de l'interaction du cerveau avec le milieu extérieur. Le premier modèle opérationnel est le perceptron simple inspiré du modèle visuel et capable d'apprentissage. Il a été proposé en 1958 par Rosenblatt. Les limites du Perceptron monocouche du point de vue performance ont été montrées en 1969 par les mathématiciens Minsky et Papert. Les travaux de Hopfield en 1982 ont montrés que des réseaux de neurones artificiels étaient capables de résoudre des problèmes d'optimisation et ceux de Kohonen (1982) ont montré qu'ils étaient capables des résoudre des tâches de classification et de reconnaissance [13].

4.4.2 Neurone biologique :

Le système nerveux est composé de milliards de cellules : c'est un réseau de neurones biologiques. En effet, les neurones ne sont pas indépendants les uns des autres, ils établissent entre eux des liaisons et forment des réseaux plus ou moins complexes.

Le neurone biologique est composé de trois parties principales :

Le corps cellulaire composé du centre de contrôle traitant les informations reçues par les dendrites.

Les dendrites sont les principaux fils conducteurs par lesquels transitent l'information venue de l'extérieur.

L'axone est fil conducteur qui conduit le signal de sortie du corps cellulaire vers d'autres neurones [17].

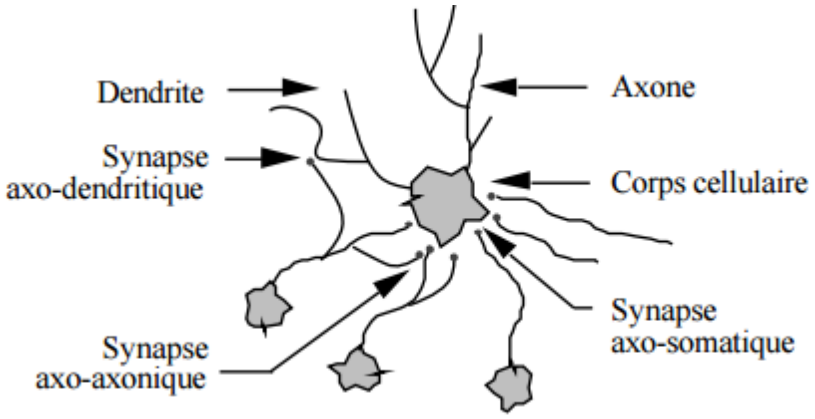


Figure 9 : Structure d'un neurone biologique[18]

4.4.3 Neurone formel :

Neurone formel ou artificiel c'est un model mathématique simplifié d'un neurone biologique, il possède des entrées , des poids appelées des coefficients synaptique et calcule la somme pondérer des entrées moyennant les poids synaptique, met ensuite un signal de sortie moyennant une fonction appelée fonction seuil.

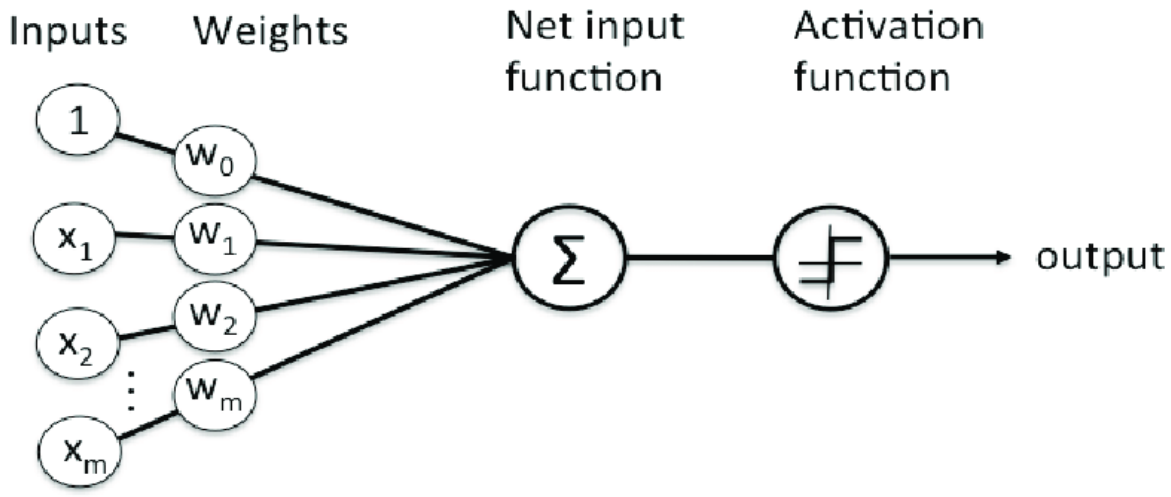


Figure 10 : Structure d'un neurone artificiel [16]

4.4.4 Fonction d’agrégation :

Il existe plusieurs mais les plus courantes sont: la somme pondérés et le calcule de distance .

La somme pondérée consiste à calculer la somme de toutes les entrées multipliées par leur poids : $\sum_{i=1}^{n} Ei * Wi$

Le calcul des distances consiste plutôt à comparer les entrées aux poids et calculer la distance entre les deux , la distance et donnée par :$\sum_{i=1}^{n} (Ei - Wi)^2$

4.4.5 Fonction d’activation

Il s'agit d'une fonction qui permet de transformer le signal entrant dans une unité (neurone) en signal de sortie (réponse). Généralement, les fonctions d'activation ont un effet "d'aplatissement". Les fonctions d'activation les plus courantes sont : la fonction tangente hyperbolique, logistique sigmoïde, exponentielle [14][15].

Binary Step Function

La fonction de pas binaire peut être utilisée comme fonction d'activation lors de la création d'un classificateur binaire, ou elle permet d’obtenir des sorties binaires, 1 si la valeur agrégée calculée est plus grande que le seuil, 0 sinon.

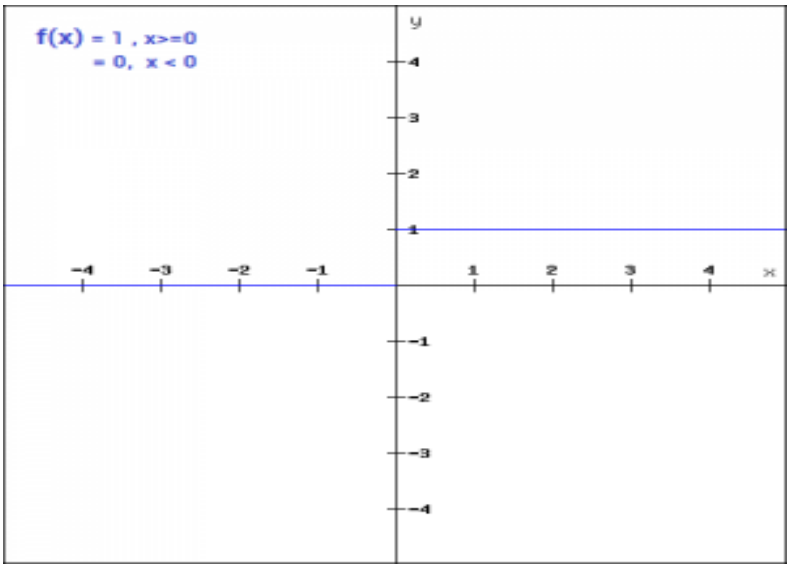


Figure 11: Tracé de Binary step function

Sigmoïde :

C'est l'une des fonctions d'activation non linéaires les plus utilisées. La sigmoïde transforme les valeurs comprises entre 0 et 1. Voici l'expression mathématique de la sigmoïde :

f(x) = 1/(1+e^-x)

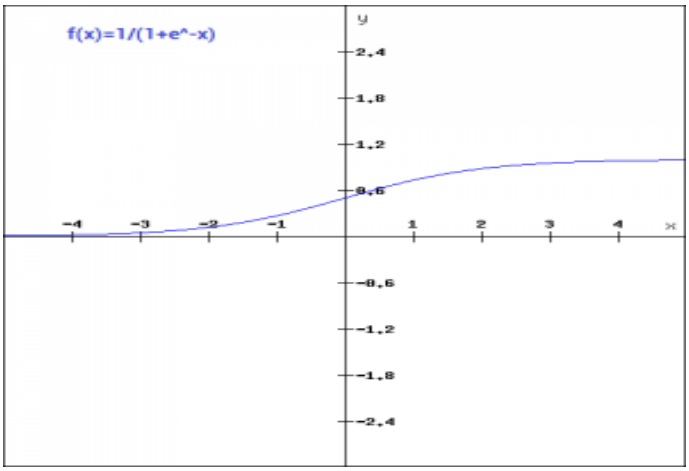


Figure 12 : Tracé de sigmoide

Relu :

La fonction d'activation linéaire rectifiée, ou ReLU en abrégé, est une fonction linéaire par morceaux qui produit directement l'entrée si elle est positive, sinon elle produit zéro. Elle est devenu la fonction d'activation par défaut pour de nombreux types de réseaux neuronaux, car un modèle qui l'utilise est plus facile à former et obtient souvent de meilleures performances.

afin d'utiliser la descente de gradient stochastique avec rétro-propagation des erreurs pour former des réseaux neuronaux profonds, il faut une fonction d'activation qui ressemble et se comporte comme une fonction linéaire, mais qui est en fait une fonction non linéaire permettant d'apprendre des relations complexes dans les données..

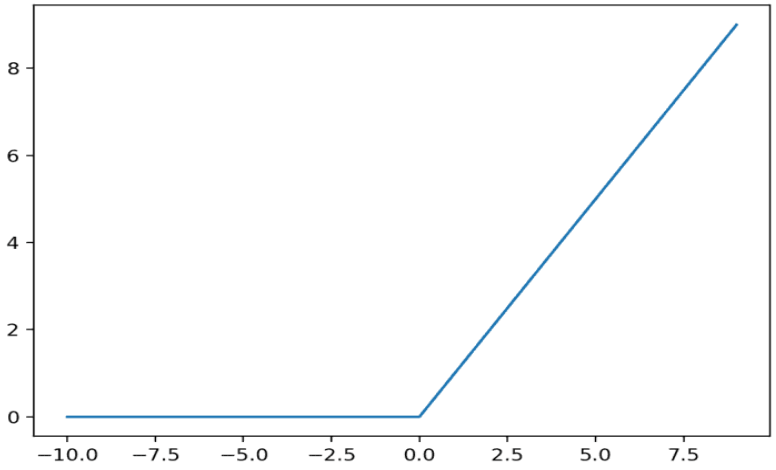


Figure 13 : Tracé de RELU pour les entrées négatives et positives

La dérivée de la fonction linéaire rectifiée est également facile à calculer. Rappelons que la dérivée de la fonction d'activation est nécessaire lors de la mise à jour des poids d'un nœud dans le cadre de la rétro-propagation de l'erreur.

La dérivée de la fonction est la pente. La pente pour les valeurs négatives est de 0,0 et la pente pour les valeurs positives est de 1,0.

5 MLP (Multi-layer Perceptron)

Un perceptron multicouche (MLP) est un réseau neuronal artificiel profond. Il est composé de plus d'un perceptron. Ils sont composés d'une couche d'entrée pour recevoir le signal, d'une couche de sortie qui prend une décision ou fait une prédiction sur l'entrée, et entre ces deux couches, un nombre arbitraire de couches cachées qui sont le véritable moteur de calcul du MLP. Les MLP à une couche cachée sont capables d'approximer n'importe quelle fonction continue.

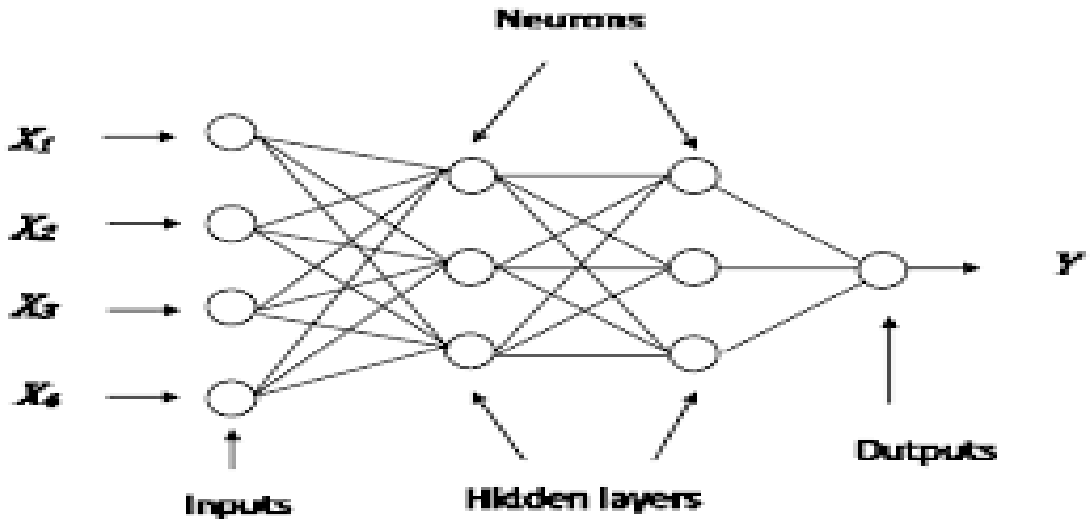


Figure 14 : réseaux neurones avec 2 couches cachés

Les perceptrons multicouches sont souvent appliqués à des problèmes d'apprentissage supervisé : ils s'entraînent sur un ensemble de paires d'entrées et de sorties et apprennent à modéliser la corrélation (ou les dépendances) entre ces entrées et ces sorties. L'apprentissage consiste à ajuster les paramètres, ou les poids et les biais, du modèle a fin de minimiser l'erreur. La rétro-propagation est utilisée pour effectuer ces ajustements de poids et de biais par rapport à l'erreur, et l'erreur elle-même peut être mesurée de diverses manières, notamment par l'erreur quadratique moyenne (RMSE).

Dans le passage avant, le flux de signaux passe de la couche d'entrée à la couche de sortie en passant par les couches cachées, et la décision de la couche de sortie est mesurée par rapport aux étiquettes de la vérité de base.

Dans le passage en arrière, en utilisant la rétro-propagation et la règle de la chaîne de calcul, les dérivées partielles de la fonction d'erreur en fonction des différents poids et biais sont rétro propagées à travers le MLP. Cet acte de différenciation nous donne un gradient, ou un paysage d'erreur, le long duquel les paramètres peuvent être ajustés pour rapprocher le MLP du minimum d'erreur. Cela peut être fait avec n'importe quel algorithme d'optimisation basé sur le gradient, comme la descente de gradient stochastique. Le réseau continue à jouer cette partie de tennis jusqu'à ce que l'erreur ne puisse pas être inférieure. Cet état est connu sous le nom de convergence [19].

5.1 Formation des réseaux Perceptron multicouches

L'objectif du processus de formation est de trouver l'ensemble des valeurs de poids qui permettront à la sortie du réseau neuronal de correspondre le plus possible aux valeurs cibles réelles. La conception et la formation d'un réseau perceptron multicouche soulèvent plusieurs questions :

- La sélection du nombre de couches cachées à utiliser dans le réseau.
- Décider du nombre de neurones à utiliser dans chaque couche cachée.
- Trouver une solution globalement optimale qui évite les minima locaux.
- Convergence vers une solution optimale dans un laps de temps raisonnable.
- Validation du réseau neuronal pour vérifier qu'il n'y a pas d'ajustement excessif.

5.1.1 Sélection du nombre de couches cachées

Pour presque tous les problèmes, une couche cachée est suffisante. Deux couches cachées sont nécessaires pour modéliser des données présentant des discontinuités, telles qu'un modèle d'onde en dents de scie. L'utilisation de deux couches cachées améliore rarement le modèle, et peut introduire un plus grand risque de convergence vers un minimum local. Il n'y a aucune raison théorique d'utiliser plus de deux couches cachées.

5.1.2 Décider du nombre de neurones à utiliser dans les couches cachées

L'une des caractéristiques les plus importantes d'un réseau perceptron est le nombre de neurones dans la ou les couches cachées. Si le nombre de neurones est insuffisant, le réseau sera incapable de modéliser des données complexes et l'ajustement résultant sera médiocre.

Si le nombre de neurones est trop élevé, le temps d'apprentissage peut devenir excessivement long et, pire encore, le réseau peut s'adapter de manière excessive aux données. Lorsque l'ajustement excessif se produit, le réseau commence à modéliser le bruit aléatoire des données. Le résultat est que le modèle s'adapte extrêmement bien aux données d'apprentissage, mais qu'il se généralise mal aux nouvelles données non vues. La validation doit être utilisée pour tester ce phénomène. En pratique, on arrive à trouver des paramètres satisfaisants, mais il n’y a pas vraiment de cadre théorique pour formaliser tout cela [19][20].

5.1.3 Recherche d'une solution globalement optimal

Un réseau neuronal typique peut comporter quelques centaines de poids dont il faut trouver les valeurs pour produire une solution optimale. Si les réseaux neuronaux étaient des modèles linéaires comme la régression linéaire, la recherche de l'ensemble optimal de poids serait un jeu d'enfant. Mais la sortie d'un réseau neuronal en fonction des entrées est souvent très non linéaire, ce qui rend le processus d'optimisation complexe.

Les méthodes d'optimisation telles que la descente la plus abrupte et le gradient conjugué sont très susceptibles de trouver des minima locaux si elles commencent la recherche dans une vallée proche d'un minimum local. Elles n'ont pas la capacité de voir la situation dans son ensemble et de trouver le minimum global.

Plusieurs méthodes ont été essayées pour éviter les minima locaux. La plus simple consiste à essayer un certain nombre de points de départ aléatoires et à utiliser celui qui présente la meilleure valeur. Une technique plus sophistiquée appelée recuit simulé améliore cette méthode en essayant des valeurs aléatoires très éloignées les unes des autres, puis en réduisant progressivement ("refroidissement") les sauts aléatoires dans l'espoir que l'emplacement se rapproche du minimum global.

5.1.4 Convergence vers la solution optimale - Gradient conjugué

Étant donné un ensemble de valeurs de poids de départ sélectionnées aléatoirement, La plupart des algorithmes d'apprentissage suivent ce cycle pour affiner les valeurs de poids : (1) exécuter un ensemble de valeurs de variables à travers le réseau en utilisant un ensemble provisoire de poids, (2) calculer la différence entre la valeur cible prédite et la valeur cible réelle pour ce cas, (3) faire la moyenne des informations d'erreur sur l'ensemble des cas d'apprentissage, (4) propager l'erreur en arrière à travers le réseau et calculer le gradient (vecteur de dérivées) du changement d'erreur par rapport aux changements de valeurs de poids, (5) faire des ajustements aux poids pour réduire l'erreur. Chaque cycle est appelé une époque.
Comme l'information sur l'erreur est propagée vers l'arrière dans le réseau, ce type de méthode d'apprentissage est appelé rétro propagation.

5.1.5 Rétro-propagation.

L'algorithme de formation par rétro propagation a été décrit pour la première fois par Rumelhart et McClelland en 1986 ; il s'agissait de la première méthode pratique de formation des réseaux neuronaux. La procédure originale utilisait l'algorithme de descente de gradient pour ajuster les poids vers la convergence en utilisant le gradient. En raison de cette histoire, le terme "rétro propagation" ou "back-propagation" est souvent utilisé pour désigner un algorithme de formation de réseau neuronal utilisant la descente de gradient comme algorithme de base.

5.1.5.1 Comment fonctionne l'algorithme de rétro-propagation

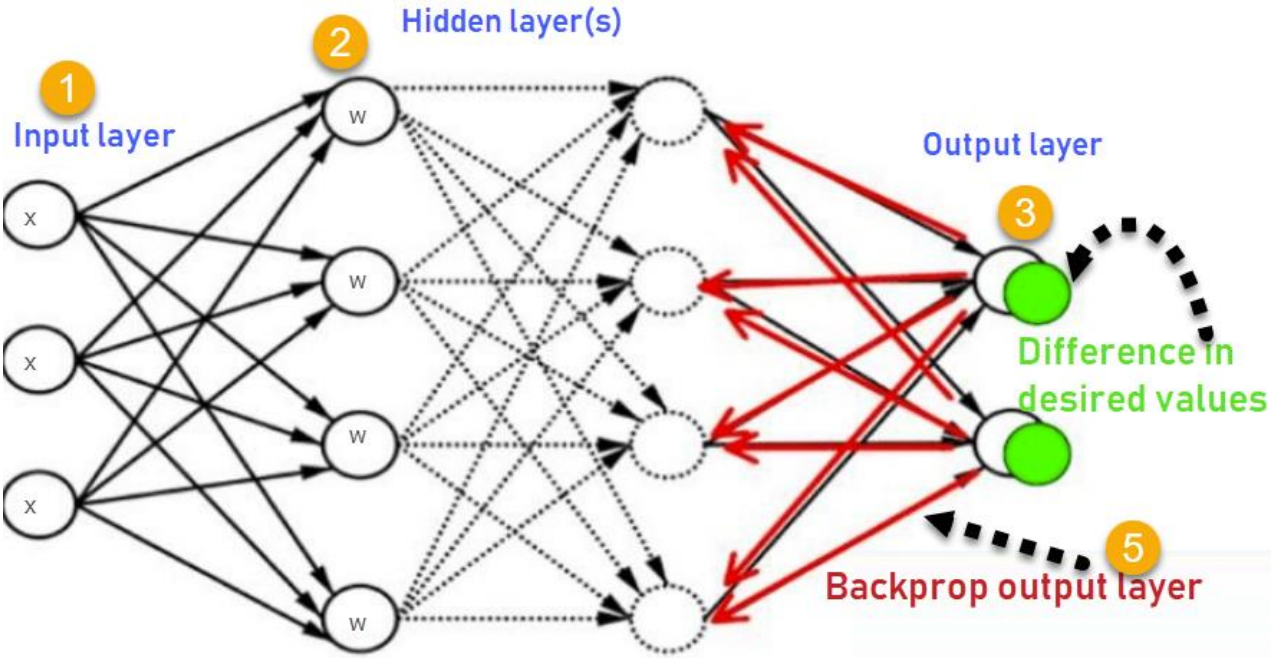


Figure 15 : Rétro-propagation dans MLP [21]

- Les entrées X, arrivent par le chemin pré-connecté
- Les entrées sont modélisées à l'aide de poids réels W. Les poids sont généralement choisis de manière aléatoire.
- Calculez la sortie de chaque neurone de la couche d'entrée, aux couches cachées, à la couche de sortie.
- Calculez l'erreur dans les sorties
- Revenez de la couche de sortie à la couche cachée pour ajuster les poids de façon à réduire l'erreur.

- Répétez le processus jusqu'à ce que la sortie souhaitée soit atteinte.

5.1.5.2 Avantages de l'utilisation de la rétro-propagation :

Les avantages les plus importants de la rétro-propagation sont les suivants :
Le back-propagation est rapide, simple et facile à programmer.
Il n'y a pas de paramètres à régler, à part le nombre d'entrées.
C'est une méthode flexible car elle ne nécessite pas de connaissances préalables sur le réseau.
C'est une méthode standard qui fonctionne généralement bien
Elle ne nécessite pas de mention particulière des caractéristiques de la fonction à apprendre [21].

5.1.5.3 Inconvénients de l'utilisation de la rétro-propagation :

La performance réelle de la rétro-propagation sur un problème spécifique dépend des données d'entrée.
L'algorithme de rétro-propagation dans l'exploration de données peut être assez sensible aux données bruyantes.

6 Conclusion

Dans ce deuxième chapitre, nous avons présenté une introduction au domaine de classification automatique des données, où nous avons abordé les notions de base de ce domaine, les différentes catégories de la classification ainsi que les différentes techniques de classification de chaque catégorie.

Nous avons aussi mis l'accent sur l'utilisation des réseaux de neurones comme outils de modélisation par apprentissage. Ces derniers permettent d'ajuster des fonctions non linéaires très générales à des ensembles de points. Comme toute méthode qui s'appuie sur des techniques statistiques, l'utilisation de réseaux de neurones nécessite que l'on dispose de données suffisamment nombreuses et représentatives, Enfin, nous avons décrit les applications des réseaux de neurones. Avant de proposer notre approche en doit d'abord présenter quelques travaux antérieurs dans le chapitre 3.

Chapitre 3 :

Etat de l'art sur les IDS

1 Introduction :

Ces dernières années, en raison de l'utilisation intensive d'internet, nombre d'ordinateur en réseau a augmenté dans votre vie quotidienne. Les faiblesses des serveurs permettent aux pirate de s'introduire dans les ordinateurs en utilisant non seulement des types d'attaques connu, mais aussi de nouveaux types d'attaques, généralement utilisent des bases de données qui contiennent des caractéristiques d'un ensemble d'attaques déjà déduites pour énoncer des menaces inconnues qui sont plus sophistiques et plus difficiles a détecter. Pour protéger les ordinateurs contre ces attaques, le système de détection d'intrusion, Dans ce chapitre, nous allons montrer quelques-unes des recherches sur l'avancement et l'efficacité des IDS existants.

2 Une étude sur les systèmes de détection d'intrusion utilisant les données de l'hôte

En 2018, Glass-Vanderlan et al [22] ont fait une étude sur les IDS qui exploitent les sources des données de l'hôte pour détecter les attaques sur le réseau d'entreprises. Les articles étudiés des modèles testés sur des ensembles des données de réseau, mais applicables aux IDS basés sur l'hôte (HIDS). Selon l'article [22], chaque IDS est composé de 3 éléments qui le construit. Le premier élément, c'est la collecte des données qui représente un ensemble de signatures des logiciels malveillants connus ou des règles élaborées par des experts. Certaines unités de ces données sont mises sous la forme d'une liste d'attributs formant ce qu'est appelé vecteur de caractéristiques, cette action de conversion en fonction de certaines caractéristiques représente le 2e élément d'un IDS.

Il y a aussi un moteur de décision qui est l'élément le plus important dans un IDS. Le moteur de décision, c'est l'algorithme qui analyse les entrées et fait une comparaison des caractéristiques des données collectées pour arriver à une décision finale du résultat si l'entrée est considérée comme une attaque ou bien non. Quelques attaques échappent généralement aux algorithmes de détection des abus vue qu'elles sont nouvelles et inconnues ou elles ne font pas partie des données d'entraînement collectées. Les bases de données utilisées par les IDS sont presque limitées aux anciennes informations et parfois de mauvaise qualité. La mise à jour régulière en temps quasi-réel de ces données est l'une des solutions proposées pour ce problème qui permet de créer et de faire connaître des ensembles de données hôte et réseau réalistes [23].

L'IDS basé sur l'hôte se situe sur le système qu'il surveille, ce qui donne une excellente visibilité de l'état du système, mais peut-être désactivé par les attaquants ayant accès au système ce qui le rend inutile. À l'inverse des systèmes NIDS qui sont indépendants des systèmes surveillés, mais rend la détection plus difficile. Le HIDS peut offrir un environnement de sécurité pour des applications individuelles où seulement cette application ou logiciel qui sera protégé, ou bien détecter des intrusions sur le système complet. Ce dernier se décrit en deux formes, HIDS avec des journaux systèmes et HIDS avec des données d'audit. Les journaux systèmes sont l'ensemble d'enregistrements et d'actions associées aux activités et événements d'un programme ou un système d'exploitation observé. La détection d'intrusions avec ces journaux nécessite beaucoup de temps processeur et exige beaucoup d'espace de stockage ce qui rend leur accessibilité et l'extraction des fonctionnalités plus difficiles. L'analyse peut se faire en se concentrant sur la précision de la détection ou bien sûr l'architecture des IDS.

Les données d'audit constituent les informations et actions de contrôle enregistrés et exécutés par le système sécurisé qui réfère aux activités des utilisateurs. Dans l'exemple de l'article [22], les journaux d'audit permettent de visualiser les connexions réseau, les actions en ligne de commande les escalades de privilèges et les modifications apportées. Les données d'appel système représentent un objet principal du noyau du système d'exploitation. Une séquence de tous les appels invoqués par un processus unique peut aussi être utilisée pour la détection, c'est ce que montrent les recherches réalisées. Les séries d'appels enregistrées peuvent être courtes ce qui peut invoquer une ignorance des données en comparaison avec les longues séries, mais elles permettent moins de calculs pendant l'analyse.

Pour une grande précision de détection avec moins de surcharge, plusieurs caractéristiques peuvent s'ajouter aux appels vus la sensibilité de la détection à la longueur de ces derniers. Ces caractéristiques sont soit séquentielles (n-Grammes) ou bien basées sur la fréquence. Des faux positifs risquent de se produire en utilisant cette approche si la charge de calcul pour la collecte augmente [23].

2.1 Modèle de corrélation d'alertes en temps réel (RACC) :

Plusieurs études ont étais réalisé pour arriver au recherche du RACC et ce sont inspirer du travail de Soleimani et al [24] qui a présente un cadre multicouche.

Le RACC est composée d'une collection de matrices, en termes plus précis, il y aura une matrice pour chaque scenario appelé code-book extrait de la base données. Au lieu de conserver toutes les alertes et hyper alertes dans le mémoire principales, cette méthodes ne conserve que les indices dans le mémoire principale, cela réduira efficacement le temps de corrélation des alertes, une fois que les indices ont cette crée dans la phase hors ligne la corrélation des alertes peut être effectuée en ligne en temps réel. L'idéal pour ces méthodes est de pouvoir prédire les prochains mouvements des utilisateurs malveillants en analysant les incidents actuels, chaque ligne doit représenter une alerte unique qui pourrait être utilise par le système, les colonnes représente le protection, chaque élément est représenté par un bit et n'occupe qu'un bit d'espace dans la mémoire principale, Chaque scénario est placé dans un vecteur, chaque vecteur contient la position de l'hyper alerte pour former un vecteur indice. Ce qui nous permet d'avoir le bon chemin d'accès au code-book, c'est pour ça qu'un arbre rouge-noir est utilisé.

2.2 Fonctionnement du modèle RACC

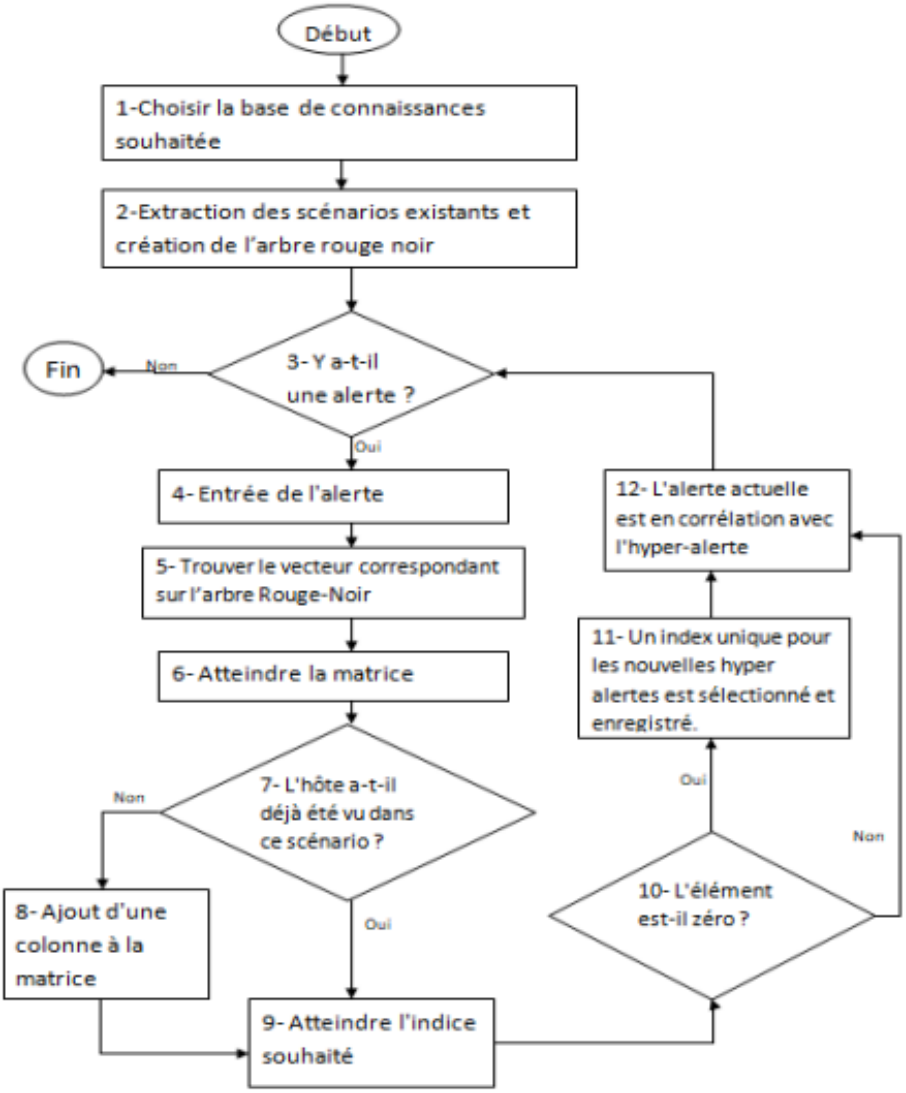


Figure 16 : Diagramme de fonctionnement du model RACC

3 Détection de la menace persistante avancée à l'aide de L'analyse de corrélation de l'apprentissage machine

L'attaque APT est une attaque persistante et ciblée sur une organisation spécifique et se déroule en plusieurs étapes. L'objectif principal d'une APT est l'espionnage puis l'exfiltration de données. Par conséquent, l'APT est considérée comme une nouvelle version plus complexe de l'attaque en plusieurs étapes. Ces APT représentent un défi pour les méthodes de détection actuelles car elles utilisent des techniques avancées et exploitent des vulnérabilités inconnues. De plus, les dommages économiques dus à une attaque APT réussie sont importants. Le coût potentiel des attaques est la principale motivation des investissements dans les systèmes de détection et de prévention des intrusions. Les APT sont actuellement l'une des menaces les plus sérieuses pour les entreprises et les gouvernements.

La plupart des recherches dans le domaine de la détection des APT se sont concentrées sur l'analyse des APT déjà identifiées ou sur la détection d'une APT particulière qui utilise un logiciel malveillant spécifique. Certains travaux ont tenté de détecter de nouvelles attaques APT. Cependant, ils présentent de sérieuses lacunes en matière de détection en temps réel , de détection de toutes les étapes d'une attaque APT ,d'équilibre entre les taux de faux positifs , de faux négatifs et de corrélation d'événements. Les travaux existants sont encourageants. Cependant, la détection précise et rapide des APT reste un défi.

3.1 Architecture de la MLAPT

Sur la base de ce raisonnement, l'architecture du système proposé (MLAPT) est présentée à la figure 17 La MLAPT a trois phases principales : détection des menaces, corrélation des alertes et prédiction des attaques.

Dans un premier temps, le trafic réseau est analysé et traité afin de détecter les éventuelles techniques utilisées dans le cycle de vie d'une APT. À cette fin, huit modules de détection ont été développés, chaque module met en œuvre une méthode pour détecter une technique utilisée dans une des étapes de l'attaque APT, et il est indépendant des autres.

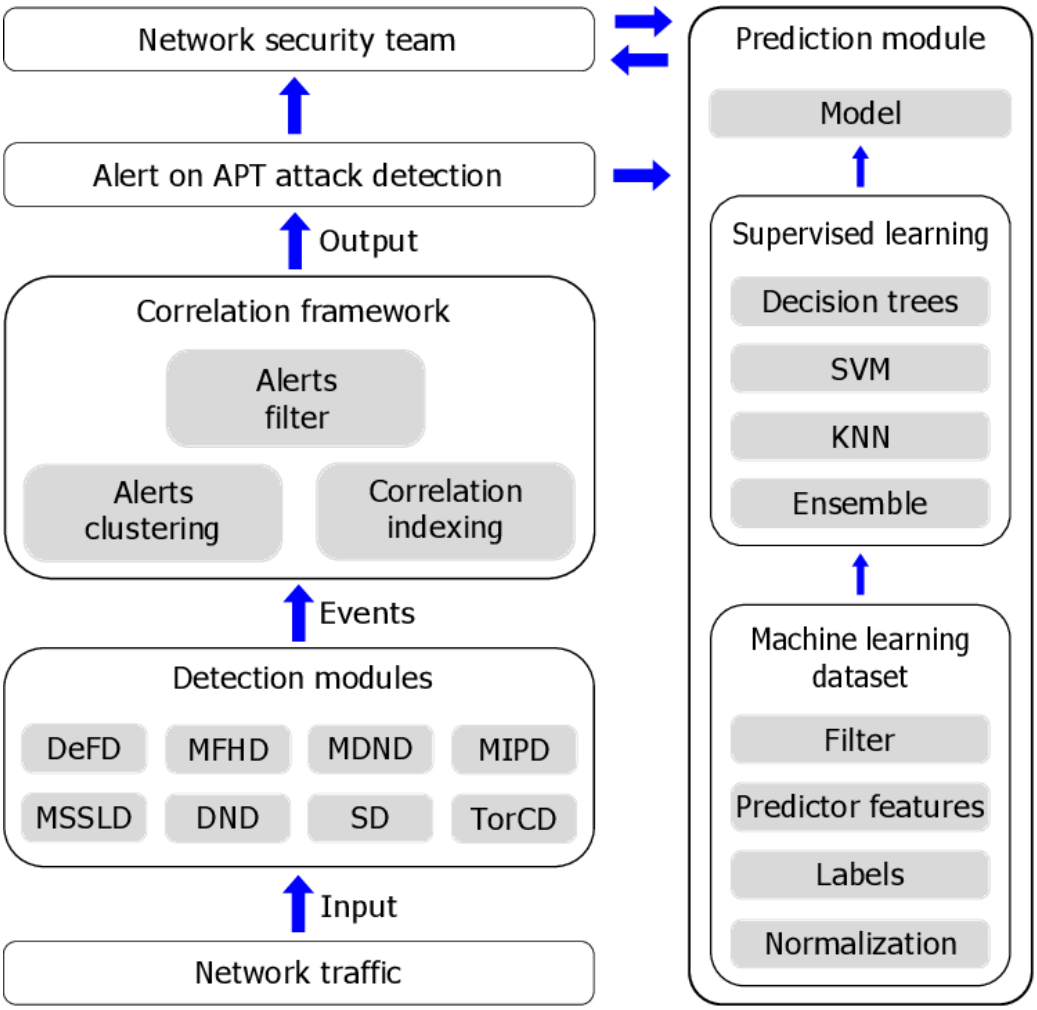


Figure 17 : L'architecture de la MLAPT [25]

Les alertes émises par les modules de détection individuels sont ensuite transmises au cadre de corrélation. L'objectif du cadre de corrélation est de trouver les alertes qui pourraient être liées et appartenir à un même scénario d'attaque APT. Le processus de cette phase comporte trois étapes principales : le filtrage des alertes pour identifier les alertes redondantes ou répétées , le regroupement des alertes qui appartiennent très probablement au même scénario d'attaque APT, et l'indexation de la corrélation pour évaluer le degré de corrélation entre les alertes de chaque groupe.

Dans la phase finale, un module de prédiction basé sur l'apprentissage automatique est utilisé par l'équipe de sécurité du réseau pour déterminer la probabilité que les premières alertes développent une attaque APT complète. Cela permet à l'équipe de sécurité du réseau de prévoir l'attaque APT dans ses premières étapes et d'appliquer la procédure requise pour l'arrêter avant son achèvement et minimiser les dommages. La détection d'une attaque APT est différente de la prédiction. La détection peut se faire lorsque deux ou plusieurs étapes de l'APT sont corrélées. Cependant, la prédiction peut être réalisée après que les deux premières étapes de l'APT soient liées.

3.2 Comparaison des performances entre l'approche étudiée et les systèmes de détection APT existants

Tableau 1 : Une comparaison entre le MLAPT et d'autres systèmes existants [23]

APT detection system	Autonomy	APT steps	speed	TPR	FPR	Prediction accuracy
MLAPT	Autonomous	4	Real time	81.8%	4.5%	84.8%
TerminAPTor	Agent-based	4	Real time	100%	high	No
C&C-based	Autonomous	1	Off-line	83.3%	0%	No
Spear phishing based	Autonomous	1	Real time	97.2%	14.2%	No
Context-based	Agent-based	4	Real time	?	27.88%	No

En utilisant l'ensemble de données de corrélation, qui est la sortie du cadre de corrélation FCI sur une période de six mois, l'ensemble de données d'apprentissage automatique est préparé.

Comme il n'existe pas d'algorithme d'apprentissage automatique qui puisse être considéré comme le meilleur ou l'optimal, plusieurs expériences doivent être réalisées sur l'ensemble de données d'apprentissage automatique en utilisant plusieurs algorithmes d'apprentissage automatique, puis une comparaison entre les modèles formés est effectuée.

L'application Classification Learner de Matlab est utilisée pour former des modèles de classification de l'ensemble de données d'apprentissage automatique. L'entraînement automatisé est effectué pour rechercher le meilleur type de modèle de classification, y compris les arbres de décision, les machines à vecteurs de support, les voisins les plus proches et la classification d'ensemble. La validation croisée est utilisée comme schéma de validation pour examiner la précision de prédiction de chaque modèle formé.

La validation croisée est une technique d'évaluation de modèle utilisée pour évaluer la performance d'un algorithme d'apprentissage automatique en faisant des prédictions sur de nouveaux ensembles de données qui n'ont pas été formés. Pour ce faire, on partitionne un ensemble de données et on utilise un sous-ensemble pour former l'algorithme et le reste des données pour les tests. Chaque cycle de validation croisée implique la partition aléatoire de l'ensemble de données original en un ensemble d'entraînement et un ensemble de test. L'ensemble d'apprentissage est ensuite utilisé pour former un algorithme d'apprentissage supervisé et l'ensemble de test est utilisé pour évaluer ses performances. Ce processus est répété plusieurs fois et la précision moyenne est utilisée comme indicateur de performance. Le tableau 5 montre la précision de prédiction pour tous les algorithmes de classification étudiés utilisés pour former les modèles de classification.

Les résultats expérimentaux montrent que le meilleur algorithme de classification est le SVM linéaire, avec une précision de prédiction de 84,8 %. Ce modèle formé peut être sauvegardé par l'équipe de sécurité du réseau pour être appliqué au trafic en temps réel lorsqu'une nouvelle alerte

4 Conclusion

Dans ce chapitre, nous avons abordé quelques méthodes de détection avec une brève explication de leur fonctionnement et la sophistication et la variété des cyber-attaques, y compris les attaques APT, augmentent de manière exponentielle à l'échelle mondiale. Il y a un besoin urgent de développer un système efficace de détection rapide et précise des attaques pour une réponse et une défense rapides. Cet article a développé un nouveau système basé sur l'apprentissage automatique (MLAPT) pour détecter et prédire les attaques APT dans une approche holistique. Dans le chapitre 4 nous allons proposer notre approche ainsi son évaluation.

Chapitre 4 :

Optimisation et algorithmes arbre de décision

1 Introduction :

Notre application se base sur le modèle arbre de décision déjà décrit dans le chapitre précédent pour faire une classification supervisée des connexions TCP/IP de la base NSL_KDD afin d'établir un système de détection d'intrusions basé sur l'analyse du comportement de ces connexions et permet de les classifier en deux types (attaque et normale). Pour cela nous avons montré dans ce chapitre les différentes phases du développement de notre application, en commençant par une description de la base de données NSL-KDD et les étapes de prétraitement que nous avons fait sur cette dernière, ensuite nous présentons notre modèle de classification suivi par la discussion des résultats obtenus.

2 Description de la base NSL-KDD :[29]

La base NSL-KDD (Network Security Layer-Knowledge Discovery in Databases) a été fondé sur l'ensemble de données KDD99, Cette dernière est une base de données qui contient des connexions TCP /IP extraites de l'ensemble de données d'évaluation des systèmes de détection d'intrusions. KDD99 été réalisées en 1998 par l'agence de L'armé américain DARPA (Défense Advanced Research Projects Agency) et AFRL (Laboratoire de recherche de l'armée de l'air), ensuite MIT Lincoln Labs8 a collecté et distribué les ensembles de données pour l'évaluation du système de détection d'intrusions de réseau informatique.

La base NSL-KDD est un ensemble de données qui représente une version réduite de l'originale KDD 99,proposé en 2010 par les chercheurs dans le domaine de détection d'intrusions réseaux afin de résoudre certains problèmes qui ont apparu dans la base KDD 99. NSL-KDD considérée comme un ensemble de données de référence pour aider les chercheurs à comparer les différentes méthodes de détection d'intrusions. Le NSL-KDD présente les différences suivantes par rapport à l'originale KDD 99:

- Il n'inclut pas les enregistrements redondants dans les données d'apprentissage, ce qui améliore la performance de classification.

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

• Le nombre d'enregistrements dans les données d'apprentissage et les ensembles d'essais sont raisonnables, ce qui il est abordable d'exécuter les expériences sur l'ensemble complet sans la nécessité de choisissez au hasard une petite portion. Par conséquent, les résultats d'évaluation des travaux de recherche seront cohérents et comparables.

Les données NSL-KDD contiennent des enregistrements de connexion TCP/IP, dont chaque enregistrement est constitué de 41 attributs caractérisant la connexion, et un attribut représente 5 classes qui sont : normales et 4 types d'attaques.

Les principaux types d'attaques de l'ensemble de données NSL-KDD sont (Probing, Dénis de Services, User to Root, Remote to User).

3 contenue de l'ensemble de données NSL-KDD :[26]

KDDTrain + .ARFF: Ensemble complet NSL-KDD avec étiquettes binaires en format ARFF

KDDTrain + .TXT: Ensemble complet de trains NSL-KDD incluant les étiquettes d'attaque et le niveau de difficulté au format CSV

KDDTrain + _20Percent.ARFF: Un sous-ensemble de 20% du fichier KDDTrain + .arff

KDDTrain + _20Percent.TXT: Un sous-ensemble de 20% du fichier KDDTrain + .txt

KDDTest + .ARFF: Le test complet NSL-KDD avec des étiquettes binaires au format ARFF

KDDTest + .TXT: Ensemble de test complet NSL-KDD incluant les étiquettes d'attaque et le niveau de difficulté au format CSV

KDDTest-21.ARFF: Un sous-ensemble du fichier KDDTest + .arff qui n'inclut pas les enregistrements avec un niveau de difficulté de 21 sur 21

KDDTest-21.TXT: Sous-ensemble du fichier KDDTest+ .txt qui n'inclut pas les enregistrements ayant un niveau de difficulté de 21 sur 21

3.1 Distribution des connexions réseau de NSL KDDTest+, et NSL KDDTrain_20%[28]

Table 2 : distribution de connexion réseau de NSL KDD

Catégorie	Nombre d'enregistrement en KDDTrain_20%		Nombre d'enregistrement en KDDTest+	
Normal	13499	53.39%	9711	43.08%
Dos	9234	36.65%	7458	33.08%
Probe	2289	9.09%	2421	10.74%
R2L	209	0.83%	2754	12.22%
U2R	11	0.04%	200	0.88%
Nombre total des enregistrements	25192		22544	

3.2 Attributs de la base NSL-KDD : [29]

Ces attributs peuvent être classés en trois groupes :

- Les attributs de base: ces attributs décrivent les informations de base d'une connexion, telles que la durée, les hôtes source et destination, port et flag.
- Les attributs du trafic: ces attributs sont basés sur des statistiques, tels que le nombre de connexions vers la même machine.
- Les caractéristiques du contenu: ces attributs sont construits à partir de la charge utile (Data) des paquets du trafic tels que nombre d'échec de connexion et le nombre d'accès aux fichiers de contrôle.

4 Processus de génération du modèle de classification :

L'élaboration de notre modèle respecte les phases
principales suivantes : le prétraitement, l'apprentissage et la phase de test.

4.1 Prétraitement de l'ensemble de données de la base NSL-KDD :

Les Données de NSL-KDD sont de trois types: numérique, Nominale et binaire. Les attributs 2, 3 et 4 sont nominales, 7, 12, 14, 15, 21 et 22 sont binaires, et le reste des attributs sont de type numérique, avant de passer au travail expérimental, l'ensemble de données NSL-KDD est d'abord passé par une opération de prétraitement des données et la conversion du type des attributs en suivant les étapes décrites dans ce qui suit

4.1.1 Numérisation (conversion des données symboliques vers numériques):

Comme nous l'avons dit précédemment la base de données de NSL-KDD contient 3 attributs ("protocol_type", "service" et "flag".) ont des données nominales. Sachant que les modèles de réseaux de neurones n'acceptent que des attributs numériques. Nous avons converti les trois attributs mentionnées ci-dessus vers des données numériques, Il existe plusieurs méthodes de conversion, parmi lesquelles la conversion alphabétique simple que nous avons choisi pour numériser les attributs de type nominal de la base de données NSL-KDD

4.1.2 Conversion alphabétique simple:

La conversion simple consiste à remplacer les valeurs des données catégoriques en ordre alphabétique par des nombres, Les données de l'attribut "type_protocole" par exemple, contiennent trois valeurs catégorielles distinctes : "tcp", "icmp" et "udp". Ces valeurs sont d'abord classées par ordre alphabétique puis on les attribuant un nombre pour chaque catégorie distinct de valeurs [29] .

4.1.2.1 La Conversion alphabétique simple des valeurs de l'attribut "protocol_type"[29]

Table 3 Conversion de l'attribut "protoco_type" [29]

Avant conversion	Apres conversion
Icmp	0
Tcp	1
Udp	2

4.1.2.2 La Conversion alphabétique simple des valeurs de l'attribut "service" :

Pour l'attribut "service" qui contient 70 catégories distincts de valeurs, Après l'ordonnancement alphabétique de ces catégories, la valeur de donnée "aol" sera remplacé par "0", "auth" par "1" et "bgp" par "2" et on continue la conversion selon l'ordre alphabétique

Table 4 Conversion de l'attribut "service"

Avant conversion	Après Conversion	Avant conversion	Après Conversion	Avant conversion	Après Conversion
Aol	0	http_443	23	printer	46
Auth	1	http_8001	24	private	47
Bgp	2	imap4	25	red_i	48
courier	3	IRC	26	remote_job	49
csnet_ns	4	iso_tsap	27	rje	50
Ctf	5	Klogin	28	shell	51
daytime	6	Kshell	29	smtp	52
discard	7	Ldap	30	sql_net	53
domain	8	Link	31	ssh	54
domain_u	9	Login	32	sunrpc	55
Echo	10	Mtp	33	supdup	56
eco_i	11	Name	34	Systat	57
ecr_i	12	netbios_dgm	35	Telnet	58
Efs	13	netbios_ns	36	Tftp_u	59
Exec	14	netbios_ssn	37	Tim_i	60
finger	15	Netstat	38	Time	61
ftp	16	Nnsp	39	urh_i	62
ftp_data	17	nntp	40	urp_i	63
gopher	18	ntp_u	41	uucp	64
harvest	19	Other	42	uucp_path	65
hostnames	20	pm_dump	43	vmnet	66
http	21	pop_2	44	whois	67
http_2784	22	pop_3	45	X11	68
Z39_50				69	

4.1.2.3 La Conversion alphabétique simple des valeurs de l'attribut "flag"

Table 5 Conversion de l'attribut "flag"

Avant conversion	Après Conversion	Avant conversion	Après Conversion
OTH	0	S1	6
REJ	1	S2	7
RSTO	2	S3	8
RSTOS0	3	SF	9
RSTR	4	SH	10
S0		5	

4.1.3 Normalisation

La normalisation est une méthode de prétraitement des données qui permet de réduire la complexité des modèles. C'est également un préalable à l'application de certains algorithmes**.** La normalisation standardise la moyenne et l'écart-type de tout type de distribution de données, ce qui permet de simplifier le problème d'apprentissage en s'affranchissant de ces deux paramètres.

Pour effectuer cette transformation, on utilise une fonction de normalisation Min Max suivante :

$$nouvval=\left(\frac{\text{ancval} - \text{Minanc}}{\text{Maxanc} - \text{Minanc}}\right)\times (Max_{nouv} - Min_{nouv}) + Min_{nouv}$$

4.1.4 Extraction de caractéristiques :[28]

Une fois le prétraitement des données terminé, il est généralement impossible d'introduire directement les données dans un apprenant en raison de leurs dimensions élevées, et il est donc nécessaire de sélectionner un petit nombre de caractéristiques utiles pour l'apprentissage automatique.

Afin d'accélérer l'algorithme d'apprentissage ultérieur, dans les 122 données caractéristiques à dimensions obtenues ci-dessus, les principales caractéristiques de DoS, Probe, R2L et U2R ont été trouvées en réduisant les dimensions ou en extrayant des caractéristiques. Un nouveau DT-RFE est utilisé. Ici, RFE consiste à construire itérativement le modèle, puis à choisir les meilleures (ou les pires) caractéristiques qui sont sélectionnées en fonction des coefficients.

Ensuite, les caractéristiques sélectionnées sont mises de côté, et le processus se répète pour les autres caractéristiques. Le processus continue jusqu'à ce que toutes les caractéristiques soient parcourues. Les séquences éliminées dans ce processus sont les caractéristiques d'ordre. La caractéristique du RFE lui-même nous permet de mieux effectuer la sélection manuelle des caractéristiques. La stabilité du RFE dépend largement du modèle utilisé en bas de l'itération. Si la relation entre une caractéristique et une variable de réponse est non linéaire, on peut utiliser une méthode arborescente ou un modèle linéaire étendu. En général, certaines méthodes arborescentes sont plus faciles à utiliser, car elles modélisent des relations non linéaires et ne nécessitent pas beaucoup de débogage. Dans l'extraction de caractéristiques, le modèle sous-jacent est sélectionné comme un simple algorithme d'arbre de décision. En outre, l'entropie de l'information est un indice important de la sélection des caractéristiques dans l'arbre de décision. L'arbre de décision calcule l'entropie de l'information de l'ensemble de données et divise l'ensemble de données couche par couche. Enfin, chaque type d'échantillon est divisé séparément. L'entropie est la mesure de l'incertitude d'une variable aléatoire. Supposons que X soit une variable aléatoire avec un nombre fini de valeurs, et que sa distribution de probabilité soit notée comme suit :$P(X=xi)= pi.$

LA méthode REF utilise l'arbre de décision pour la formation de plusieurs cycles. Selon les coefficients de pondération générés par la formation, les meilleures caractéristiques sont conservées pour le tour de formation suivant. Pour les modèles de prédiction avec des coefficients de pondération des caractéristiques, la méthode RFE réduit de manière récursive la taille de l'ensemble des caractéristiques examinées pour sélectionner les caractéristiques. Tout d'abord, sur la base des caractéristiques originales, les modèles de prédiction sont entraînés à attribuer un poids à chaque caractéristique. Ensuite, l'ensemble de caractéristiques peut être simplifié en supprimant les caractéristiques dont le poids est le plus faible. Cette récurssion se poursuivra jusqu'à ce que le nombre de caractéristiques restantes atteigne le nombre requis Par rapport à RFE, RFECV ajoute un processus de validation croisée pour mieux sélectionner le nombre optimal de caractéristiques. Pour un ensemble de caractéristiques avec d, le nombre de tous ses sous-ensembles est 2d-1 (y compris l'ensemble vide). L'arbre de décision calcule l'erreur de validation de tous les sous-ensembles et sélectionne le sous-ensemble ayant la plus petite erreur comme caractéristique sélectionnée [29].

4.1.5 Apprentissage et établissement du modèle de classification

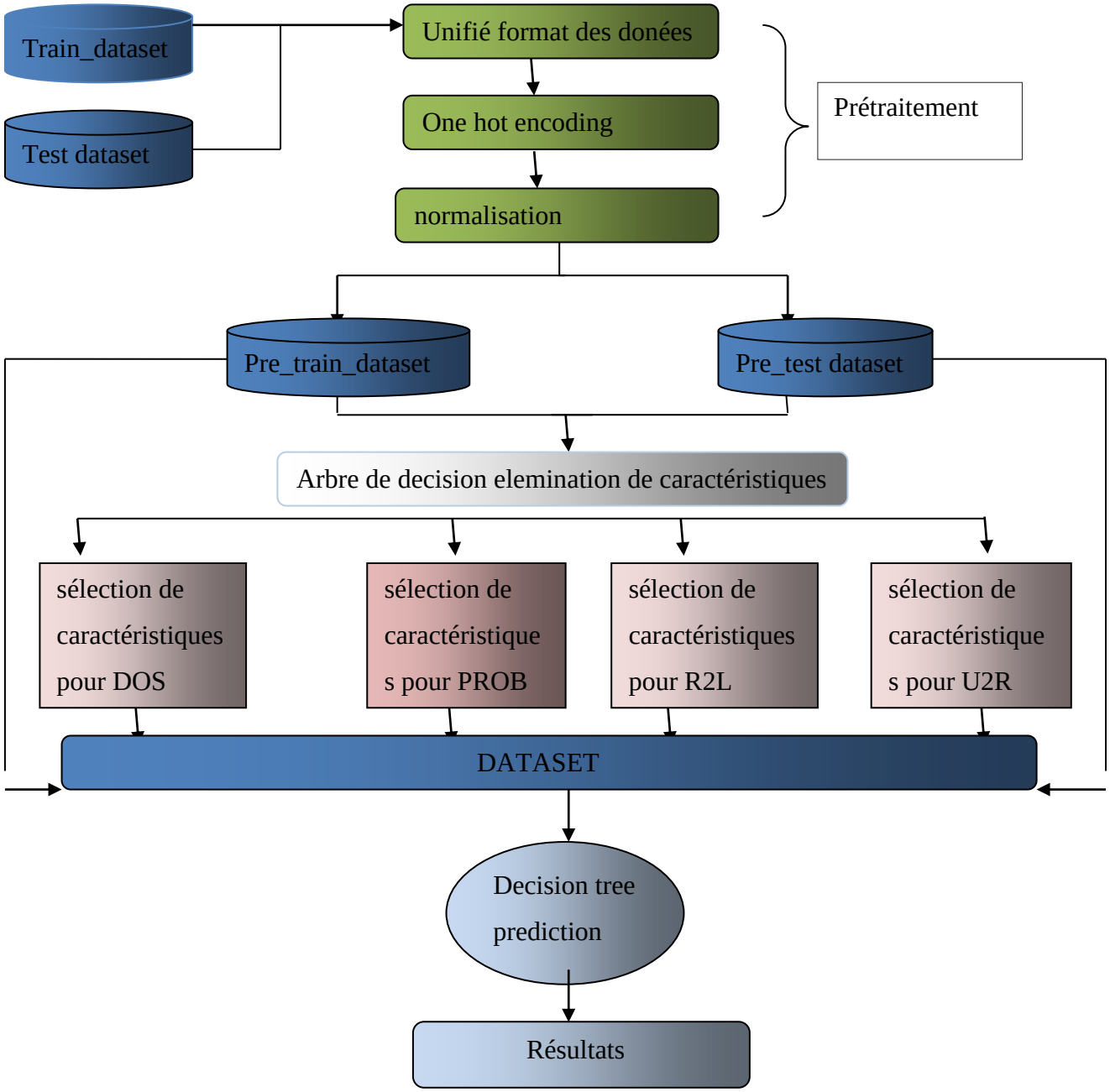


Figure 18 Diagramme de l'approche propose

L'algorithme de fusion par empilement en combinant DT-RFE est proposé de manière créative, et sa mise en œuvre se compose de trois étapes. Tout d'abord, l'ensemble de données doit être préparé et normalisé .Ensuite, une extraction de caractéristiques pour la réduction de la dimension est construite.

Après l'extraction des caractéristiques, un algorithme d'apprentissage automatique est utilisé pour classifier et vérifier l'ensemble de données, et enfin l'apprentissage d'ensemble est utilisé pour générer une fonction générique

Un système de détection d'intrusion basé sur un arbre de décision (DT) est implémenté pour évaluer la performance du système

sur la base de données de détection d'intrusion bien connus (NSL-KDD). L'arbre de décision est une approche non paramétrique et non linéaire utilisée pour l'apprentissage de la classification supervisée et la régression.

Les étapes de la méthode proposée sont présentées dans la **figure 10**. Dans la première étape, le jeu de données est sélectionné à partir du dépôt de données de NSL-KDD.

Après cela, un prétraitement est appliqué aux données sélectionnées pour convertir les attributs non numériques en attributs numériques.

Dans l'étape suivante, la méthode de sélection des caractéristiques est appliquée pour réduire la dimensionnalité. L'échantillonnage aléatoire est appliqué pour se débarrasser des biais de formation.

Les trois étapes suivantes définissent l'algorithme CART :

L'induction de l'arbre, l'élagage de l'arbre et le test de l'arbre résultant avec un nouvel ensemble de données de test. CART est une méthode de partitionnement récursif qui construit un arbre de décision binaire en le divisant à chaque nœud. Le nœud racine de l'arbre de décision contient échantillons d'entraînement complets. Dans la dernière étape, la performance est évaluée en utilisant divers paramètres de performance telle que la précision de la classification, le taux de détection et le taux de faux positifs.

5 Expérimentation et analyse :

La clé pour mesurer la qualité d'un système de détection d'intrusion est de savoir s'il a un taux de détection élevé et un taux de FP faible. Par conséquent, afin d'évaluer la performance du modèle de détection d'intrusion proposé, dans ce document, l'exactitude (ACC), la précision (PRE), le taux de détection (DR), et le F1-Score sont sélectionnés. Le taux de précision est utilisé pour mesurer la précision de la classification. Le taux de détection est utilisé pour mesurer le pourcentage de comportements anormaux qui sont détectés. La précision est utilisée pour mesurer combien de cas d'utilisation dans les résultats du test sont des comportements anormaux. En outre, le FP est utilisé pour juger de manière exhaustive DR et ACC.

Les formules de calcul correspondantes sont présentées ci-dessous :

$$ACC = \frac{TP + TN}{TP + TN + FP + FN},$$

$$PRE = \frac{TP}{TP + FP},$$

$$DR = \frac{TP}{TP + FN},$$

$$f_1 = \frac{2 * DR * PRE}{DR + PRE},$$

Où True Positive (TP) est le nombre d'échantillons positifs correctement identifiés, True Négative (TN) est le nombre d'échantillons positifs correctement identifiés, False Positive (FP) est le nombre d'échantillons positifs identifiés par erreur, et False Négative (FN) est le nombre d'échantillons négatifs identifiés par erreur.

Chaque étape de RFECV donne un nombre spécifique de caractéristiques de l'ACC de sorte que le nombre minimum de caractéristiques puisse être obtenu lorsque l'ACC atteint un maximum. En même temps, RFECV peut sélectionner les caractéristiques choisies. Par conséquent, RFECV est utilisé pour sélectionner le nombre optimal de caractéristiques.

Les caractéristiques de la sélection finale de DoS, Probe, R2L et U2R sont présentées dans le tableau :

Table 1 Caractéristique de DoS, Probe, R2L, and U2R on NSL-KDD

Caractéristique de DoS, Probe, R2L, and U2R on NSL-KDD	
DoS	'dst_host_same_srv_rate'; 'dst_host_serror_rate'; 'num_compromised'; 'same_srv_rate'; 'diff_srv_rate'; 'dst_host_count'; 'dst_host_srv_serror_rate'; 'ecr_i'; 'RSTR'; 'wrong_fragment'; 'dst_bytes'; 'src_bytes.'
Prob	'src_bytes'; 'telnet'; 'smtp'; 'private'; 'http'; 'ftp_data'; 'finger'; 'dst_host_rerror_rate'; 'dst_host_same_src_port_rate'; 'dst_host_diff_srv_rate'; 'dst_host_same_srv_rate'; 'rerror_rate'; 'dst_bytes.
R2L	'duration'; 'imap4'; 'ftp_data'; 'dst_host_srv_diff_host_rate'; 'dst_host_same_src_port_rate'; 'dst_host_same_srv_rate'; 'dst_host_srv_count'; 'dst_host_count'; 'num_access_files'; 'num_failed_logins'; 'hot'; 'dst_bytes'; 'src_bytes.'
U2R	'duration'; 'dst_host_srv_diff_host_rate'; 'dst_host_count'; 'srv_count'; 'num_shells'; 'num_file_creations'; 'root_shell'; 'dst_bytes'; 'dst_host_same_srv_rate'; 'hot'; 'src_bytes.'

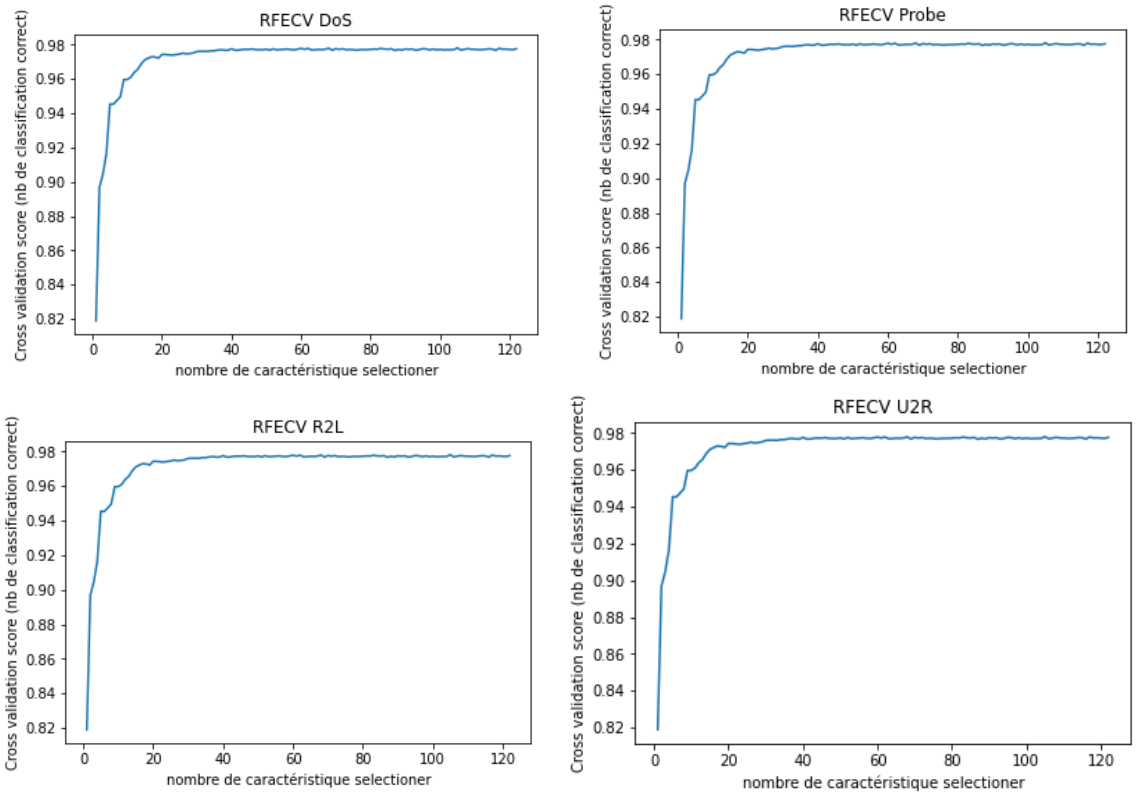


Figure 19 Nombres de caractéristiques actuellement sélectionnées par RFECV

L'axe des x de la figure 20 représente le nombre de caractéristiques actuellement sélectionnées par RFECV dans le processus récursif, et l'axe des y représente le score de validation croisée sous le nombre. DT-RFE calcule le classement de l'importance du résultat de toutes les caractéristiques sous l'objectif actuel. On peut voir que différents nombres optimaux de caractéristiques sont nécessaires pour la prédiction de différents types d'attaques. En d'autres termes, la prédiction de différents types de cibles nécessite des caractéristiques différentes pour obtenir de meilleurs résultats. Avec le classement et le nombre optimal de caractéristiques, nous pouvons obtenir les caractéristiques requises comme le tableau 5. Comme le montre la figure 18, nous menons des expériences sur NSL-KDD. En ce qui concerne le résultat de la sélection des caractéristiques, nous prenons NSL-KDD comme exemple pour le montrer et comme le montre le Tableau 5.

Tout d'abord, l'apprentissage des caractéristiques est effectué en utilisant l'ensemble de formation marqué comme entrée de l'IDS. Après plusieurs expériences, les paramètres optimaux sont sélectionnés et enregistrés, et le modèle IDS formé est retenu. Ensuite, les données restantes forment un ensemble de données de test pour mesurer la précision de détection que le modèle réalise sur chaque attaque.

```
Accuracy: 0.99663 (+/- 0.00259)
Precision: 0.86481 (+/- 0.08952)
Recall: 0.91672 (+/- 0.10661)
F-measure: 0.88628 (+/- 0.07462)
```

Figure 20 Métriques dévaluation pour DOS

```
Accuracy: 0.99639 (+/- 0.00341)
Precision: 0.99505 (+/- 0.00477)
Recall: 0.99665 (+/- 0.00483)
F-measure: 0.99585 (+/- 0.00392)
```

Figure 21 les Métriques dévaluation pour PROB

```
Accuracy: 0.99571 (+/- 0.00328)
Precision: 0.99391 (+/- 0.00684)
Recall: 0.99267 (+/- 0.00405)
F-measure: 0.99329 (+/- 0.00512)
```

Figure 22 les Métriques dévaluation pour R2L

```
Accuracy: 0.97920 (+/- 0.01054)
Precision: 0.97150 (+/- 0.01741)
Recall: 0.96957 (+/- 0.01381)
F-measure: 0.97050 (+/- 0.01482)
```

Figure 23 les Métriques dévaluation pour U2R

6 Conclusion:

Nous avons abordé dans ce dernier chapitre la base de données NSL-KDD qui a été utilisée pour l'apprentissage et test du modèle de détection d'intrusions généré basé sur les réseaux de neurones artificiels. Ainsi, un nouveau système intelligent de détection d'intrusion basé sur l'empilement est proposé, et il utilise un algorithme DT-RFE pour extraire moins de caractéristiques. Notre méthode permet d'améliorer et d'optimiser l'ensemble de données et d'augmenter l'utilisation des ressources en supprimant les enregistrements non corrélés et redondants. Lorsque la précision d'un seul modèle d'apprentissage automatique est difficile à améliorer, l'empilement peut être utilisé pour empiler les modèles d'apprentissage automatique afin d'améliorer la précision. Dans l'ensemble, notre méthode présente un DR plus élevé et un FPR plus faible, ainsi qu'une meilleure précision de reconnaissance pour chaque type d'attaque.

CONCLUSION GENERAL

Conclusion général :

Les attaques informatiques sont en forte hausse ces dernières années et représentent aujourd'hui un risque réel qui menace les réseaux informatiques, les applicatifs et les systèmes d'information des entreprises. En d'autres termes, La découverte périodique et permanente de ces attaques, en temps opportun peut contribuer à les réduire en prenant les moyens de protection nécessaires, Cela nous a dirigés, dans ce mémoire, vers le développement d'un modèle du système de détection d'intrusions comme moyen d'identification des attaques, dans le but d'éviter leurs dommages. Pour réaliser ce modèle nous nous sommes basé sur les réseaux de neurones artificiels, Ces derniers sont utiles pour la simulation de n'importe quel problème difficile à décrire avec des modèles physiques et mathématiques en raison de la capacité des réseaux de neurones d'apprendre par des exemples.

Nous avons aussi utilisé la base de données du benchmark NSL-KDD comme source de données, et nous avons fait un modèle de classification binaire (deux classes : normale et attaque) pour classifier les connexions TCP/IP en deux classes : attaque ou normal, à l'aide de perceptron multicouches(MLP) qui est le modèle de réseaux de neurones le plus approprié pour la classification des données non linéairementséparables.

En effet, le réglage des paramètres du réseau de neurones a une grande importance pour obtenir de bons résultats. Pour cette raison, Nous avons effectué plusieurs expériences afin de choisir les meilleurs paramètres (poids initiaux, pas d'apprentissage, nombre de couches et nombre de neurones dans chaque couche) qui nous donnent les meilleurs résultats en termes de taux de réussite. Nous avons aussi implémenté dans notre application une fonction qui sélectionne les meilleurs attributs en fonction de leurs gains (sélection des attributs ayant un gain supérieur ou égal à un seuil défini par l'utilisateur).Puisque les résultats sont proches avec ou sans sélection des attributs, l'utilisation d'un sous ensemble d'attributs pendant l'apprentissage du MLP nous permet de gagner le temps de calcul.

Nous avons aussi utilisé l'algorithme arbre de décision combiné avec l'algorithme de sélection de caractéristiques comme moyen d'optimisation des valeurs des poids générés par le réseau de neurones (MLP), Cela nous a conduits à des améliorations considérables particulièrement en termes de taux de réussite

Pour conclure, la majorité des objectifs tracés dans de ce travail ont été atteints, mais il reste toujours des perspectives et des améliorations possibles qui peuvent encore être réalisé dans le future, telles que :

- La réalisation d'un modèle de classification multi-classes (Normal, DOS, R2L, U2R et Probe) au lieu de la classification binaire (Normal, Attaque) réalisée dans ce travail.
- élargir l'algorithme DT-RFE pour la sélection des meilleurs paramètres d'apprentissage (nombres de couches cachés, nombre de neurones par couche, taux d'apprentissage, etc.).
- Aussi de penser à la création d'un modèle capable de capturer le trafic réseau en temps réel sans avoir besoin d'utiliser l'ensemble de données NSL KDD Test.

Bibliographie

[1] Lerman Liran, « Les systèmes de détection d'intrusion basés sur du machine Learning » Mémoire de Master, état de l'art. L'UNIVERSITE LIBRE DE BRUXELLES, Faculté des Sciences, Département d'Informatique. 2010.

[2] Ludovic Mé, « Détection des intrusions dans les systèmes d'information : la nécessaire prise en compte des caractéristiques du système surveillé », Juin 2003

[3] BOUROUH Mouloud et KANOUN Zakaria.« Détection d'intrusions à base des réseaux de neurones et algorithmes génétiques », mémoire de fin d'études ,Telemcen, Algerie(2017)

[4] Introduction aux IDS « http://www-igm.univ-mlv.fr/~dr/XPOSE2004/IDS/IDSPres.html » Consulté le 10 juillet 2021

[5] A. Mercier,(05 février 2018.) « *L'information face à l'intelligence artificielle : promesses et dangers* », [consulté enligne https://larevuedesmedias.ina.fr/linformation-face-lintelligence-artificielle-promesses-et-dangers, le 13 juin 2021] .

[6] A. Géron, « *Machine Learning avec Scikit-Learn* », Dunod : Paris, 2017.320 PAGE

[7] I. Bellin, N. Vayatis, J. Audiffren, S. Peignier A. Nicolai . (Fevrier 2019) « *Apprentissage par renforcement* »,[consulté enligne :https://dataanalyticspost.com/Lexique/apprentissage-par-renforcement, le 14 juin 2021] ;

[8] Apprentissage par renforcement : une IA puissante dans toujours plus de domaines https://hellofuture.orange.com/fr/apprentissage-par-renforcement-une-ia-puissante-dans-toujours-plus-de-domaines/ consulté le 10 juin 2021

[9] J. Brownlee, (December 2017) « *Difference Between Classification and Regression in Machine Learning* »,[consulter en ligne : https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/, le 14 juin 2021)

[10] Antti Ajanki.(28 May 2007) "Example of k-nearest neighbour classification " , sur le site https://commons.wikimedia.org/wiki/File:KnnClassification.svg consulté le 10 aout 2021.

[11] Daniel Nelson .(aout 2020) , «What is a KNN (K-Nearest Neighbors)? ».[consulter enligne: https://www.unite.ai/author/danielnelson/ le 15 juin 2021]

[12] Rohith Gandhi, (juin 2018).«Support Vector Machin, Introduction to machine learning algorithme»[consulter enligne :https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 le 15juin 2021]

[13] Anderson, J. A., 1995, Introduction to Neural Networks (Cambridge, MA: MIT Press).

[14] K.S.Narendra and K.Parthazarathy,"identification and control of dynamical systems using neural networks"IEEE Transactions on Neural Networks ,vol. 1, no.1, pp .4-25,1990

[15] J. Lagarde, « Apprentissage : Hebb », 2013.

[16] Nicole Dalia .Cilia Luca Tonetti, "Wired Bodies New Perspectives on the Machine-Organism Analogy "CNR Edizioni Italy, 2017 ,pp 151

[17] Thomas B (juin 2020) « Réseau de neurones – Deep Learning : Biologiques ou artificiels »[consulter enligne https://datascientest.com/deep-learning-reseau-de-neurones-biologiques-ou-artificiels : le «3 juillet 2021]

[18] Claude Touzet. LES RESEAUX DE NEURONES ARTIFICIELS, INTRODUCTION AU CONNEXIONNISME : COURS, EXERCICES ET TRAVAUX PRATIQUES. EC2, 1992, Collection de l'EERIE, N. Giambiasi. ffhal-01338010f

[19] Phillip H. Sherrod "multi layer perceptron" https://www.dtreg.com/solution/multilayer-perceptron-neural-networks consulté le 03 aout 2021

[20] Camacho Olmedo**,** M.T, Paegelow**,** M., Mas, J.F**.,** Escobar**,** F Geomatic Approaches for Modeling Land Change Scenarios,2018

[21] Daniel jhonson , "Back propagation neural networck :what is backpropagation algorithme in Machine Learning" https://www.guru99.com/backpropogation-neural-network.html consulté le 12aout 2021

[22] T. R. Glass-Vanderlan et al. A Survey of Intrusion Detection Systems Leveraging Host Data. arXiv :1805.06070, 2018.

[23] M. NASSOU, D. SAADI. Systèmes de détection d'intrusions et machine learning, 2020

[24] M. Soleimani et A. A. Ghorbani. Multilayer episode filtering for the multi-step attack detection. Computer Communications, vol. 35, no. 11, pp. 1368-1379, 2012.

[25] I. Ghafira et al. Detection of Advanced Persistent Threat Using Machine-Learning Correlation Analysis. Future Generation Computer Systems, Vol.89, pp.349-359, 2018.

[26] Lasitha Hiranya . Most Important things of "NSL-KDD" data set,2020

[27] NSL-KDD dataset https://www.unb.ca/cic/datasets/nsl.html consulté le 10 mai 2021

[28] K.Meriem, E.David, A.Kamal. "new wrapper feature selection algorithme for anomaly-based intrusion detection system", springer nature Switzerland AG , 2021

[29] G.Nicolescu et al.(Eds) : FPS 2020 , LNCS 12637 , pp 3-19 , 2021

Résumé

Le développement technologique, l'utilisation d'Internet à grande échelle et l'augmentation des moyens de stockage et de l'échange d'information sont contribué à un nombre élevé de cyber-attaques, qui exploitent les failles des systèmes d'information et les points faibles des systèmes de protection contre les intrusions, ce qui rend le processus de sécurisation de ces informations très important. Les spécialistes du domaine de la sécurité informatique s'occupent d'élaborer les outils nécessaires pour assurer la sécurité de l'information en développant de nouvelles technologies basées sur des moyens d'intelligence artificielle afin de détecter les pénétrations non découvertes par des dispositifs ordinaires. Dans ce contexte, nous visons à travers ce travail à réaliser un modèle de détection d'intrusions qui s'appuis sur l'apprentissage du fonctionnement " normal " des informations échangées et de signaler les divergences par rapport à ce fonctionnement de référence comme des attaques en se basant sur l'algorithme arbre de décision et la base de données de référence NSL-KDD pour générer et évaluer le modèle proposé.

***Mots clés** : intrusions, classification, réseaux de neurones, intelligence artificiel, NSL-KDD.*

Abstract

The large-scale technological, development and use of the Internet and the increase in the means of storage and exchange of information have contributed to a high number of cyber attacks, which exploit the gaps in information systems and The weak points of protection systems against intrusions, which makes the process of securing this information very important. The security specialists are making the necessary tools to ensure information security by developing new technologies based on artificial intelligence to detect the penetrations undiscovered by ordinary devices. In this context, we aim, through this work, to realize an intrusion detection model based on the learning of the "normal" functioning of the exchanged information and to signal the divergences from this reference functioning as attacks. Based on the decision tree algorithm and the NSL-KDD reference database to generate and evaluate the proposedmodel.

***Keywords:** intrusions, classification, neural networks, artificial intelligence, NSLKDD*