RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Abderrahmane Mira - Bejaia
Faculté des Sciences Exactes
Departement d'informatique

Mémoire de fin de Cycle

En vue de l'obtention d'un master recherche en informatique

Option : Intelligence Artificielle

Thème

Systèmes de détection d'intrusions et machine learning
Cas : Détection des spams

Réalisé par :

M^lle NASSOU Meriem **M^lle SAADI Djohra**

Soutenue le 27 octobre 2020 devant le jury composé de :

Présidente	**Dr K. OUAZINE**	**M.A.B U.A/Mira Bejaia, Algerie.**
Examinateur	**Mr R. OUZEGGANE**	**M.A.A U.A/Mira Bejaia, Algerie.**
Promotrice	**Dr L. HAMZA**	**M.C.A U.A/Mira Bejaia, Algerie.**

Promotion 2019/2020

- Dédicaces -

Je dédie ce modeste travail aux personnes chères à mon cœur.
À mes parents, qui ont su m'inculquer une éducation de sagesse et de claire voyance.
À mes frères Yacine, Sofiane et Jugurta.
À mes sœurs Soria et Ouezna.
À mes neveux Yanis, Salas et Abd Allah.
À mes grands-parents ,mes tantes ,mes oncles et leurs enfants.
À ma tante Farida et ses enfants Amel, Kenza et Abdarezak.
À mes amis Nacera, Cylia, Yacine et Linda qui m'ont soutenu tout où cours de réalisation de ce travail.
À ma binôme Meriem.

Djohra

Je dédie ce modeste travail aux personnes chères à mon cœur.
À mes parents, qui ont su m'inculquer une éducation de sagesse et de claire voyance.
À mon frère Mourad.
À ma sœur Anissa.
À mes tante Saida et Nassima.
À mes neveux Kader, Ramzi, Sérine.
À mes amis Lamine, Linda, Yacine, Nacira et Cylia qui m'ont soutenu tout où cours de réalisation de ce travail.
À ma binôme Djohra.

Meriem

- Remerciements -

Au terme de notre travail, on remercie Dieu tout puissant de nous avoir donné le courage et la patience pour réaliser ce modeste travail.

La réalisation de ce mémoire a été un parcours jalonné de nombreuses rencontres, sans lesquelles ce travail n'aurait pas pu aboutir. On n'aurait pas éprouvé autant de plaisir à réaliser ce travail sans ces personnes, qui par leur générosité, leur disponibilité, leur bonne humeur et l'intérêt manifesté à l'égard de notre recherche, ont grandement contribué à l'amélioration de notre travail.

On tient particulièrement à adresser nos remerciements d'abord à notre encadrant madame HAMZA lamia de nous avoir orienté durant l'élaboration de ce travail, elle a toujours été disponible à l'écoute de nos nombreuses questions.

On tient à remercier les membres du jury qui nous font le grand honneur d'évaluer ce travail.

Enfin, nos remerciements s'adressent à toutes nos familles qui sont toujours là pour nous soutenir. Et les personnes qui ont contribué de près ou de loin à la réalisation de ce modeste travail.

Djohra et Meriem

Table des matières

Table des figures

Liste des abréviations

ANN	**A**rtificial **N**eural **N**etwork
APT	**A**dvanced **P**ersistent **T**hreats
BSD	**B**erkeley **S**oftware **D**istribution
CPU	**C**entral **P**roccessing **U**nit
CSS	**C**ascading **S**tyle **S**heets
CSV	**C**omma **S**eparated **V**alues
DARPA	**D**efense **A**dvanced **R**esearch **P**rojects **A**gency
DDoS	**D**istributed **D**enial **O**f **S**ervice
DNS	**D**omain **N**ame **S**ystem
HIDS	**H**ost-Based **I**ntrusion **D**etection **S**ystem
HINFO	**H**ost **INFO**rmatio
HIDS	**H**ost-Based **I**ntrusion **D**etection **S**ystem
HTML	**H**yper**T**ext **M**arkup**L**anguage
IDS	**I**ntrusion **D**etection **S**ystems
IDWG	**I**ntrusion **D**etection exchange format **W**orking **G**roup
KDD	**K**nowledge **D**iscovery in **D**ata bases
KNN	**K**- **N**earest **N**eighbours
k-PPV	**K**-**P**lus **P**roches **V**oisins
MLAPT	**M**achine-**L**earning **A**dvanced **P**ersistent **T**hreats
NB	**N**aïf **B**ayes
NIDS	**N**etwork-Based **I**ntrusion **D**etection **S**ystem
NLTK	**N**atural **L**anguage **T**ool**K**it
RACC	**R**eal-time **A**lert **C**orrelation method based on **C**ode-books
SNORT	**S**upersonie **N**aval **O**rdnance **R**esearch **T**rack
SVM	**S**upport **V**ector **M**achine
XML	e**X**tensible **M**arkup **L**anguage

Introduction générale

Les systèmes informatiques sont devenues des outils indispensables au fonctionnement des entreprises ou tout secteurs professionnels. L'information gérée par ces systèmes peut être exposée à des attaques qui exploitent des éléments vulnérables du système d'information ce qui a mené à l'utilisation des systèmes de détection d'intrusions (IDS : Intrusion Detection Systems). Nous dénombrons à l'heure actuelle environ une centaine de systèmes de détections d'intrusions qui sont devenus pratiquement indispensables dû à l'incessant accroissement en nombre et en dangerosité des attaques depuis quelques années.

En quelques années, l'intelligence artificielle a pris une place importante dans de nombreux domaines et s'est imposée comme une technologie incontournable. Elle est devenue inséparable de notre quotidien vue sa capacité de gérer les problèmes et générer plusieurs solutions fiables et plus acceptables. Pour améliorer les capacités de détection d'intrusions, des recherches ont été réalisé montrant une combinaison nécessaire entre la sécurité et le Machine Learning où ce dernier représente la technologie qui permet aux machines d'apprendre seules à partir des données fournies.

Dans notre cas, nous avons traité le problème des spams. Les spams entraînent des pertes annuelles totales s'élevant à plusieurs dizaines de milliards de dollars, d'après les recherches obtenues. Pour régler ce problème, les filtres antispam ont été créés. L'approche d'apprentissage automatique la plus connu dans le filtrage des spams est le classificateur Bayes naïf, il permet de calculer la probabilité qu'un message est un spam ou non. Dans notre étude, nous avons utilisé l'algorithme naïf bayes multinomial pour la classification des messages, où nous avons rajouté une deuxième vectorisation des messages par rapport aux anciennes méthodes, ce qui génère des résultats avec un faible taux d'erreur.

Ce mémoire est organisé de la manière suivante.

Dans le chapitre 1, nous présentons les deux notions de base, IDSs et Machine Learning et décrivons les caractéristiques qui construisent chacun d'entre eux.

Dans le chapitre 2, nous présentons quelques travaux sur la détection d'intrusion et le Machine Learning où nous avons abordé cinq articles concernant les recherche récente montrant l'importance du Machine Learning dans la détection des intrusions, ainsi représenter l'une des recherche

sur la détection des spams faite avec l'apprentissage machine.

Dans le chapitre 3, nous présentons notre proposition sur le problème des spams. Une section sur les notions de base utilisées est représentée ainsi une explication sur l'architecture et les étapes de l'implémentation du filtre antispam proposé pour enchaîner avec les résultats obtenus.

Finalement, nous concluons par une récapitulation du travail accompli et pointant quelques travaux potentiels futurs.

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

Chapitre 1

Système de détection d'intrusions et Machine Learning

1.1 Introduction

À l'époque, on associe le terme " Sécurité Informatique " à la protection des matériels informatiques contre les agressions externes physiques. Cependant, avec le développement des technologies de communication et la possibilité d'accéder à distance aux réseaux, les informaticiens étaient obligés de trouver des moyens de contrôle tel que les systèmes de détection d'intrusions pour assurer la protection des ordinateurs, des réseaux et des programmes contre toutes formes d'accès non autorisées et c'est là que la sécurité informatique prend sa définition et s'intègre dans le domaine de Machine Learning. Le Machine Learning s'appuie sûr de nombreuses disciplines des mathématiques. Il a pris une grande importance et est devenu inséparable dans les domaines de la science. Le Machine Learning peut être classé usuellement au moins en trois catégories : apprentissage supervisé, non supervisé et par renforcement.

Ce chapitre est organisé de la manière suivante : la section 1.2 définit la notion de la sécurité informatique. La section 1.3 présente le système de détection d'intrusions. La section 1.4 présente la notion du Machine Learning. Finalement, la section 1.5 conclut ce chapitre.

1.2 Sécurité informatique

C'est l'ensemble des mesures, des contrôles et des moyens mis en oeuvre pour assurer la protection des ordinateurs, des réseaux, des programmes et des données informatiques contre toute forme non autorisée ou non intentionnelle d'accès, d'utilisation, d'intrusion, de divulgation, d'altération ou de destruction [21].

1.3 Système de détection d'intrusions

Un système de détection d'intrusions (IDS : Intrusion Detection Systems) est un logiciel qui peut être sous forme d'appareil dont le but est de repérer des activités anormales ou suspectes telle qu'une faille de sécurité, une violation de règles, ou toutes actions malveillante qui a réussi à exploiter une vulnérabilité dans un système informatique et d'alerter l'administrateur ou de réagir à des menaces détectées.

1.3.1 Types d' IDSs

Pour identifier les menaces potentielles et faire face aux attaques utilisées par les pirates étant basées sur des failles réseaux ou de programmation, les IDSs impliquent la collecte et l'analyse d'information et pour cela, il existe trois types d'IDSs :

1.3.1.1 Systèmes de détection d'intrusions de type réseaux hôte (HIDS : Host-Based Intrusion Detection System)

La détection d'intrusions basée sur hôte surveille et analyse périodiquement le trafic des données où niveau de la machine ou ordinateur dans lequel il est installé. Leur fonctionnement est basé sur l'analyse des journaux système afin d'identifier toute éventuelle violation de la politique de sécurité [21]. Ils permettent d'analyser les activités de la machine contrôlée à un niveau de détail suffisant pour déterminer lequel des processus et des utilisateurs sont impliqués dans les activités malveillantes.

1.3.1.2 Systèmes de détection d'intrusions réseaux (NIDS : Network-Based Intrusion Detection System)

La détection d'intrusions basée sur le réseaux surveille et analyse en temps réel les paquets qui circulent sur un réseau pour assurer la sécurité et renvoyer une alerte en cas de détection d'intrusions.

1.3.1.3 Systèmes de détection d'intrusions hybrides

Un IDS hybride est une combinaison des caractéristiques des HIDS et NIDS. Il offre une surveillance multiple pour les réseaux et les terminaux on se basant sur une architecture distribuée. Après avoir détecté des intrusions, le tout sera centralisé dans une machine qui va recevoir toutes les alertes ce qui permet des résultats plus exacts, avoir le moins de faux positifs.

1.3.2 Architecture fonctionnelle des IDSs

Nous décrivons dans cette section les différents composants d'un IDS. La Figure 1.1 illustre les standards de communication entre ces composants selon les travaux du groupe IDWG (Intrusion Detection exchange format Working Group).

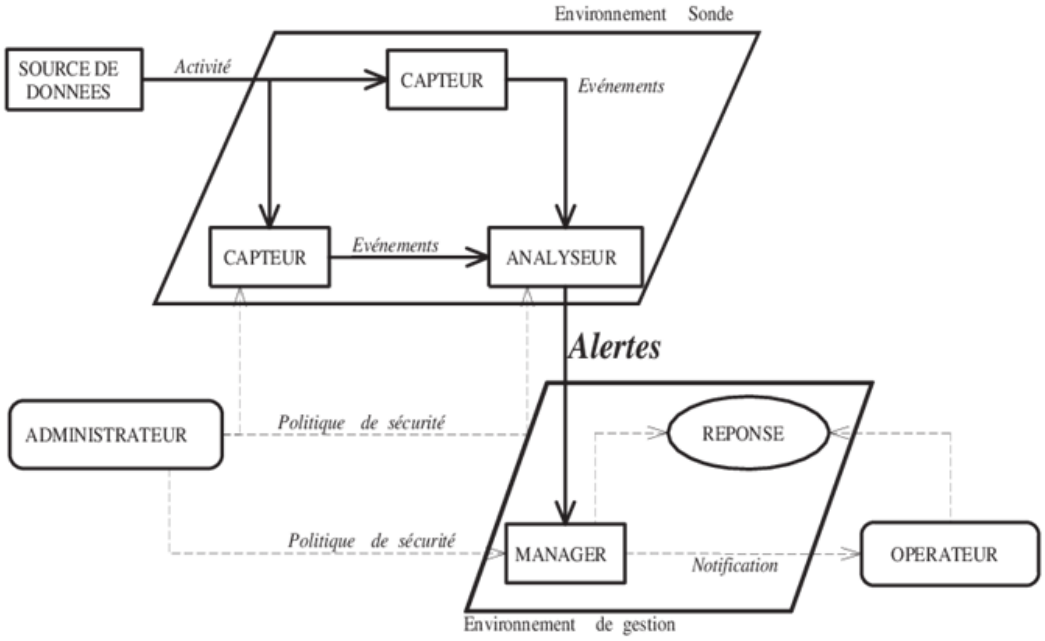

FIGURE 1.1 – Architecture IDWG d'un système de détection d'intrusions [26].

Les définitions ci-dessous sont largement inspirées de celles proposées par l'IDWG [26] :

1. **Capteur**

   Utilisé pour collecter un ensemble de données suspectes et de les envoyer vers l'analyseur sous forme d'événements.

2. **Analyseur**

   Sous forme de processus dans le rôle est de vérifier et d'attribuer une analyse profonde sur les événements reçus par le capteur concernant des signes d'activité non autorisés ou indésirables. Un ou plusieurs capteurs couplés avec un analyseur forme une sonde.

3. **Manager**

   Après l'analyse des événements, en cas de détection d'intrusions, une alerte est envoyée ou manager. Ce dernier s'occupe de la mise en forme des alertes reçus puis de notifier l'opérateur. Il peut aussi établir une réaction qui peut être un isolement de l'attaque, suppression, recouvrement ou effectuer un diagnostic dans le but de limiter ou d'arrêter l'attaque.

4. **Opérateur**

   C'est l'utilisateur principal qui surveille les sorties du système et réagit aux notifications admises pour répondre aux attaques.

5. **Administrateur**

   Personne chargée de mettre en place les lois ou la politique de sécurité qu'une sonde utilise pour la détection des menaces dans un IDS, par conséquent, de déployer et configurer les IDSs.

1.3.3 Classification des systèmes de détection d'intrusions

Les différents systèmes de détection d'intrusions disponibles peuvent être classés selon plusieurs critères qui sont [19] :

1. La méthode de détection.
2. Le comportement du système après la détection.
3. La source des données.
4. La fréquence d'utilisation.

La figure 1.2 illustre les détailles de chaque critère.

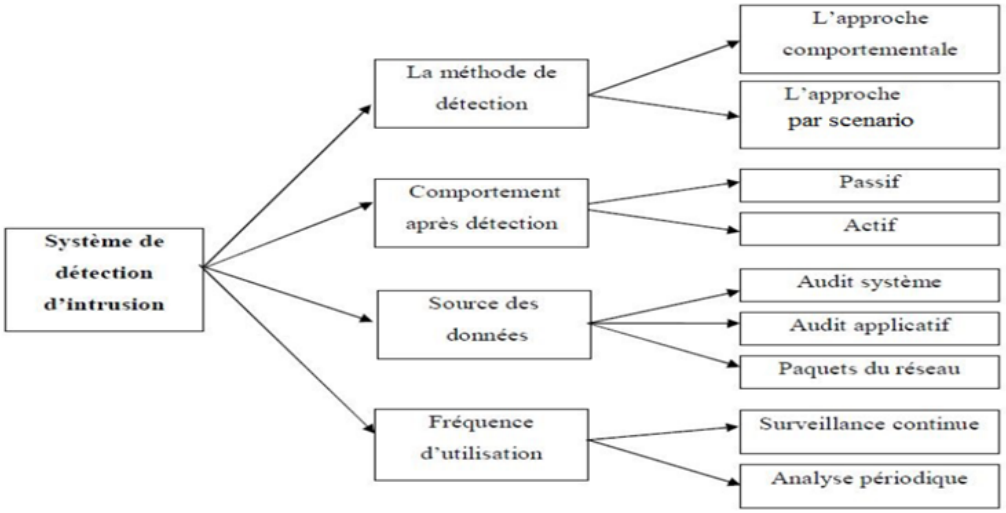

FIGURE 1.2 – Classification d'un système de détection d'intrusions [32].

1.3.3.1 Méthodes de détection des IDSs

Il existe deux approches de détection :

1. **Approche comportementale**

   Cette approche permet de détecter toute déviation par rapport au comportement normal d'un utilisateur ou d'une application (profil). Le système de détection d'intrusions compare l'activité courante au profil. Tout comportement déviant est considéré intrusif.

2. **Approche par scénario ou par signature**

   Cette technique s'appuie sur les connaissances des techniques utilisées par les attaquants contenues dans la base des scénarios d'attaque, elle compare l'activité de l'utilisateur avec la base de scénarios d'attaques, ensuite elle déclenche une alerte lors de la détection d'attaque.

1.3.3.2 Comportement après la détection d'intrusions

Il existe deux types de réponses suivant les IDSs utilisés, la réponse passive et la réponse active :

1. **Réponse passive**

   Lorsqu'une attaque est détectée, le manager génère seulement une alerte en direction de l'administrateur système par Email, message dans une console, voire même par beeper. L'administrateur prend les mesures qui s'imposent.

2. **Réponse active**

   La réponse active consiste à répondre directement à une attaque, elle implique des actions automatisées prises par un IDS qui permet de couper rapidement une connexion suspecte quand le système détecte une intrusion.

1.3.3.3 Nature des données analysées

La nature des données analysées sont composées de :

1. **Audits système**

   Utilisée par les HIDS, elle peut être une source d'information sous forme de commande système qui fournit des données sur les activités courantes du système. Elle peut aussi être un ensemble d'informations partagé par l'utilisateur durant un certain temps qui représente l'historique des événements et où finale, elle peut être un audit de sécurité dans le rôle est de collecter et associer à des utilisateurs des événements déjà définis, ce qui induit une surcharge sur le système surveillé.

2. **Audits applicatif**

   Les sources des données à analyser sont collectées à partir de l'application surveillée ce qui diminue le risque de désynchronisation entre les data qui circule entre le capteur et l'application, mais seule l'activité de l'application est surveillée.

3. **Source d'information réseaux**

   Utilisée par les NIDS, c'est des données qui circulent sur le réseau et effectuent une analyse sur le trafic. À chaque fois la charge de paquet augmente sur le réseau, le travail des capteurs devient plus difficile ce qui diminue la fiabilité et la précision du résultat d'analyse.

1.3.3.4 Fréquence d'utilisation

Représentée en deux formes :

1. **Surveillance périodique**

   Un IDS qui surveille le système d'une manière périodique veut dire faire une analyse d'un temps à un autre, à une période bien précise. Souvent retrouvé au niveau des HIDS.

2. **Surveillance en temps réel**

   Effectuer une analyse continue des données et réagir immédiatement ou attaques détectées. Souvent, retrouvée dans les réseaux où le trafic est toujours valide.

1.4 Machine Learning

Le Machine Learning a vu son apparence en 1959 par le mathématicien américain Arthur Samuel qui a développé un programme qui réussit à apprendre à jouer au dame sans aides humain. Ce dernier a défini le Machine Learning comme suis " Machine Learning is the science of getting computers to learn without being explicitly programmed " [18]. En 1998, l'Américain Tom Mitchell donna une autre définition plus avancée en énonçant qu'une machine apprend quand sa performance à faire une certaine tâche s'améliore avec de nouvelles expériences. Donc, le Machine Learning c'est la capacité d'une machine à apprendre quel calcul effectuer pour résoudre un problème donné.

Définition du Machine Learning dans le dictionnaire français

L'apprentissage automatique est le processus par lequel un algorithme évalue et améliore ses performances sans l'intervention d'un programmeur, en répétant son exécution sur des jeux de données jusqu'à obtenir, de manière régulière, des résultats pertinents.

1.4.1 Méthode d'apprentissage

Pour qu'une machine arrive à apprendre, on doit lui fournir cette capacité qui représente un ensemble de méthodes d'apprentissage inspirées de la façon dont nous, les êtres humains, apprenons à faire des choses. Le but de l'apprentissage est d'induire une fonction qui prédise les réponses associées à de nouvelles observations en commettant une erreur de prédiction la plus faible possible [16]. Parmi ces méthodes, on note : apprentissage supervisé, non supervisé et par renforcement et semi supervisé.

1.4.1.1 Apprentissage supervisé

Dans l'apprentissage supervisé, l'agent observe quelques couples types entrée-sortie et apprend une fonction de l'entrée vers la sortie [41]. C'est-à-dire, notre échantillon des données est sous forme de couple (x,y) ou x c'est l'ensemble des aspects qui représente les catégories de la cible y. À partir de cet échantillon, une fonction d'apprentissage (un modèle) sera déduite pour nous permettre d'étudier les autres cibles d'entrée non catégorisées. Apprentissage supervisé souvent utilisé pour des problèmes de reconnaissance où la machine apprend en lui donnons un ensemble d'exemples de ce qu'il faut apprendre et à partir de quelles données apprendre.

1.4.1.2 Apprentissage non-supervisé

On dit que l'apprentissage est non supervisé lorsqu'on ne connaît pas les valeurs en sortie et que l'algorithme doit travailler sur l'ensemble des aspects x où il doit reconnaître les structures communes entre ces derniers pour prédire la cible y. Dans l'apprentissage non supervisé, l'agent apprend des structures dans les données d'entrée, même s'il ne dispose pas de feedback explicite sur ses actions [41]. On peut ainsi regrouper des données dans des Clusters (c'est le Clustering), détecter des anomalies, ou encore réduire la dimension de données très riches en compilant les dimensions ensemble [42].

1.4.1.3 Apprentissage par renforcement

L'apprentissage par renforcement, c'est apprendre à agir par essais et erreur. Dans ce paradigme, un agent peut percevoir son état et effectuer des actions. Après chaque action, une récompense numérique est donnée. Le but de l'agent est de maximiser la récompense totale qu'il reçoit au cours du temps [24].

1.4.2 Processus de Machine Learning

La figure suivante permet une explication des différentes phases du processus de Machine Learning :

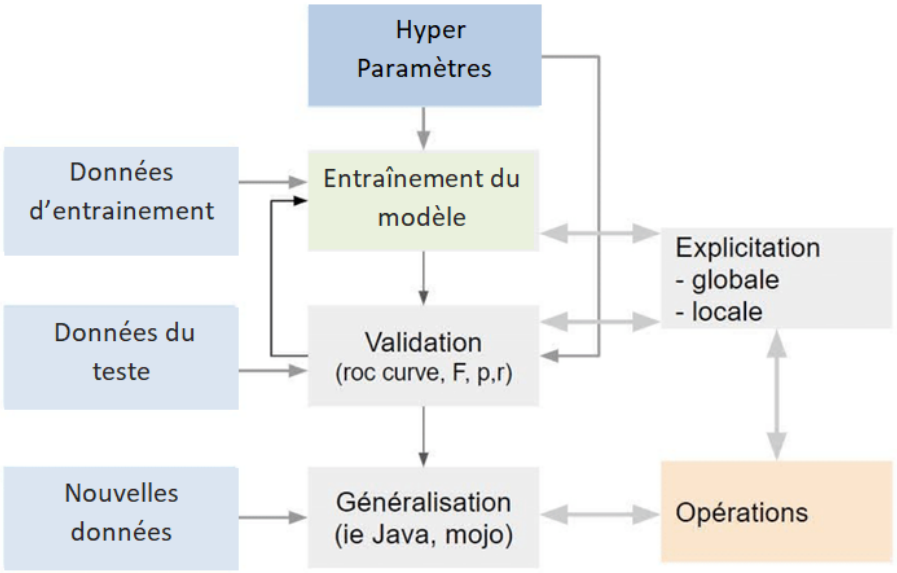

FIGURE 1.3 – Le processus de Machine Learning [18].

Afin de mieux expliquer le processus de Machine Learning, nous commençons par la définition de quelques concepts de base :

1. **Dataset**

   C'est un ensemble de données qu'on fournit à une machine sous forme d'un couple d'exemples (x, y) dans l'apprentissage supervisé où x représente les questions et y les réponses au problème que la machine doit résoudre. Dans l'apprentissage non supervisé, le dataset contient que des questions x. On ne peut pas démarrer un projet sans avoir de dataset.

2. **Modèle et ses paramètres**

   C'est une fonction mathématique qu'on développe à partir du dataset fournit et qui peut être linéaire ou bien non-linéaire. Les coefficients de cette fonction sont les paramètres du modèle et ils sont les prédicteurs x et l'annotation y que l'on veut généraliser.

3. **Les hyper-paramètres**

   Les hyper-paramètres sont les valeurs de réglage du modèle : le nombre d'itérations, les valeurs de seed (valeur aléatoire initiales), la solution initiale et les autres paramètres spécifiques des différents modèles testés [18].

4. **Fonction Coût**

   La phase de validation établit la performance du modèle en termes de taux de faux positifs (les fausses alertes) et de faux négatifs (les ratés) que l'on doit réduire simultanément [18].

Ceci se fait par une fonction coût qui représente un ensemble d'erreurs qu'un modèle nous retourne par apport à notre dataset. La validité du modèle dépend de la fonction coût à partir de laquelle la machine distingue les paramètres de notre modèle qui minimisent cette dernière.

5. **La généralisation**

   Consiste à intégrer le modèle dans un processus de big data, et dépasse l'horizon méthodologique concernant l'industrialisation des processus de plus en plus souvent assurée par une distribution du calcul [18].

Le processus de Machine Learning ce résume comme suit : Premièrement, on obtient l'échantillon d'entraînement qui représente les prédicteurs x, ce sont les données d'entraînement, et l'annotation y que l'on veut généraliser (ensemble de dataset). Puis on associe les hyper paramètres à notre modèle. Un entraînement du modèle sera effectué et passera par la phase de validation où l'efficacité du modèle s'étudiera et un calcule de nombre d'erreurs se réalisera par la fonction coût pour valider le modèle. Sachant qu'un algorithme d'apprentissage doit être choisit, une fonction issue d'un ensemble de fonction défini. Au préalable, réalise l'erreur moyenne la plus faible sur les exemples de la base d'entraînement. Enfin, on passe à la phase de généralisation où un test du modèle se déroulera sur les données globales.

1.4.3 Algorithme d'apprentissage

En Machine Learning, le traitement d'un algorithme avec des données nous permet d'avoir un modèle. On peut distinguer quatre types de modèle selon leur côté supervisé, non supervisé et par renforcement, où chaque type est basé sur un ensemble d'algorithmes.

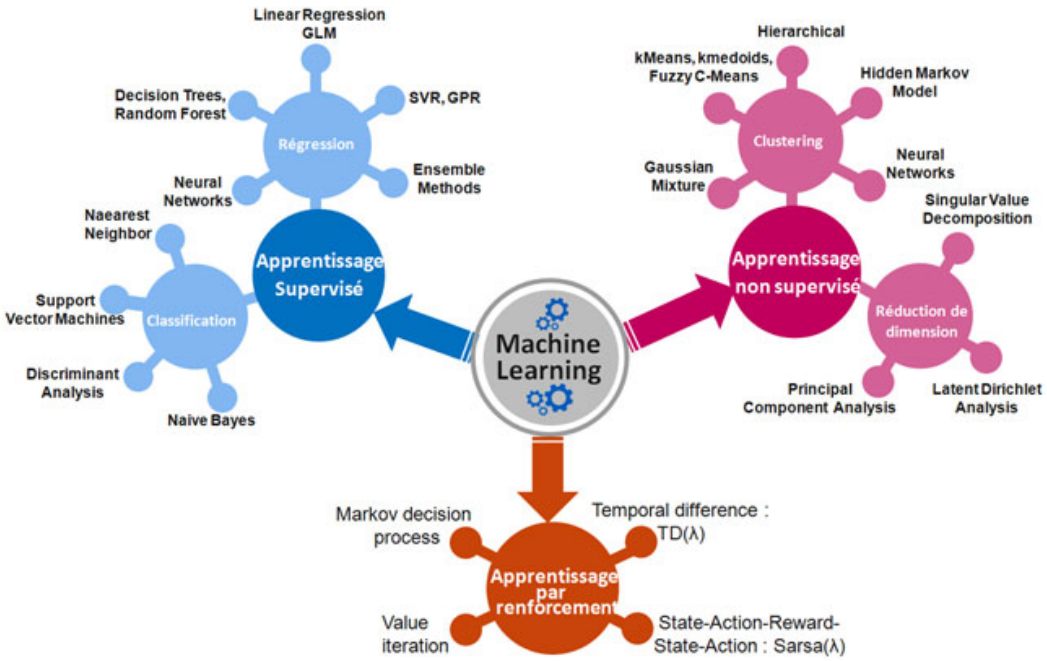

FIGURE 1.4 – Quelques algorithmes des 3 types d'apprentissage du Machine Learning : supervisé ou non et par renforcement [1].

1.4.3.1 Modèle supervisé

Quelques exemples des algorithmes d'apprentissage supervisé :

Apprentissage supervisé	Définition	Algorithmes
Classification	Affecter une étiquette connue à une nouvelle donnée **Classification binaire :** autrement dit Y = {0,1} est appelé un problème de classification binaire. **Classification multi-classe :** Un problème d'apprentissage supervisé dans lequel l'espace des étiquettes est discret et fini, autrement dit Y = {1, 2, . . . ,C} est appelé un problème de classification multi-classe. « C » est le nombre de classes.	- K plus proches voisins - Machines à vecteurs de Support. - Arbres de décision et forêts aléatoires. - Régression logistique - Naïve bayésienne.
Régression	Un problème d'apprentissage supervisé dans lequel l'espace des étiquettes est Y = R est appelé un problème de régression.	- régression linéaire - régression polynomial - régression Ridge/ lasso

FIGURE 1.5 – Algorithmes d'apprentissage supervisé [2].

Dans ce qui suit, nous présentons le principe des algorithme : K-PPV, Bayes Naïf et la régression logistique.

Classifieur K-Plus Proches Voisins (k-PPV)

Le classifieur K-plus proches voisins est un algorithme de la reconnaissance des formes, qui permet de calculer la similarité d'un nouveau cas avec les données du dataset et de prédire la classe dans laquelle il appartient selon les K plus proches voisins détectés par ordre décroissant. L'algorithme des K-plus proches voisins permet de définir une probabilité d'appartenance de la salve à chacune des classes. La probabilité est définie simplement par le rapport entre le nombre de plus proches voisins associé à la classe majoritaire divisé par le nombre total des voisins testé [37]. La valeur de K optimale peut être déterminée au préalable par la méthode du Leave-on-out qui consiste à appliquer ce classificateur à chaque vecteur de la bibliothèque pour différentes valeurs de K et à sélectionner la valeur optimale de K conduisant au pourcentage d'erreur le plus faible [37].

Bayes naïf

Le bayes naïf est un classifieur probabiliste se basant sur le théorème de bayes. Il permit de classer des éléments d'après leurs caractéristiques, on passant par une phase d'apprentissage. Les modèles sont naïf car ils assument que tous les attributs décrivant un élément à classer sont conditionnellement indépendants. A est conditionnellement indépendant de B si P(A|B,C)=P(A|C) [38].

Première formule de bayes : A et B , deux événements [38] :

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

Deuxième formule de bayes : A un événement et B un système complet d'événements [38] :

$$P(B_i|A) = \frac{P(A|B_i)P(B_i)}{P(A|B_1)P(B_1) + ... + P(A|B_n)P(B_n)}$$

Régression logistique

La régression logistique est une approche statistique qui peut être employée pour évaluer et caractériser les relations entre une variable réponse de type binaire et une, ou plusieurs, variables explicatives. Dans la régression logistique, ce n'est pas la réponse binaire (malade/pas malade) qui est directement modélisée, mais la probabilité de réalisation d'une des deux modalités (être malade par exemple) [46].

1.4.3.2 Modèle non supervisé

Quelques exemples des algorithmes d'apprentissage non supervisé :

Apprentissage non supervisé	Définition	Algorithmes
Clustering	Le clustering, ou « partitionnement » consiste à identifier des groupes dans les données. Cela permet de comprendre leurs caractéristiques générales, et éventuellement d'inférer les propriétés d'une observation en fonction du groupe auquel elle appartient.	- Réseaux neuronaux - K-means - Hiérarchique
Association	Un problème d'apprentissage de règle d'association est l'endroit ou vous souhaitez découvrir des règles décrivant une grande partie de vos données, telles que les acheteurs de x ont également tendance à acheter Y.	- A priori

FIGURE 1.6 – Algorithmes d'apprentissage non supervisé [2].

Dans ce qui suit, nous présentons le principe de l'lgorithme K-Mean Clusternig :

Algorithme de K-Mean Clustering

Le K-Mean est un algorithme de Clustering qui regroupe des données selon leur structure commune on désignant K nombre de centre de gravité et affecter chaque point du dataset à un Cluster qui contient le seul et le plus proche centre K pour qu'au final avoir un K Clusters bien centrés. L'algorithme fonctionne en 2 étapes répétées en boucle. L'étape 1 consiste à rallier chaque exemple au centre le plus proche. Après cette étape, nous avons K Clusters, l'étape 2 consiste à déplacer les centres au milieu de leur Cluster. On répète ainsi les étapes 1 et 2 en boucle jusqu'à ce que les centres ne bougent plus [42].

1.5 Conclusion

Dans ce chapitre, nous avons présenté les notions des systèmes de détection d'intrusions, cela nous a permis de constater que les IDSs sont de plus en plus fiables, d'où le fait qu'ils soient souvent intégrés dans les solutions modernes de sécurité donc ils sont indispensables aux entreprises afin d'assurer leur sécurité informatique. Nous avons aussi présenté la notion de Machine Learning et ses bases pour réussir à résoudre un problème.

Chapitre 2

Présentation de quelques travaux sur la détection d'intrusions et le Machine Learning

2.1 Introduction

Les systèmes de détection d'intrusions sont devenus indispensables dans n'importe quel appareille qui nécessite une sécurité solide et profonde soit du matériel ou des données privées. Presque toutes les entreprises utilisent ce moyen de protection depuis son apparition ou même par des individus. Cela a permis un environnement plus stable et ininterrompu, mais ça n'a pas duré pour certains vu la force que les intrus impliquent pour la destruction de ce dernier. Les intrus, généralement utilisent des bases de données qui contiennent des caractéristiques d'un ensemble d'attaques déjà déduites pour énoncer des menaces inconnues, pour cela de nouvelles méthodes de détection doivent être implémentées. Dans cette section, nous allons citer quelques-unes des recherches sur la corrélation d'alertes et la prédiction des spams qui vont nous montrer l'avancement et l'efficacité des IDSs existants.

2.2 Une étude des techniques de conception des systèmes de détection d'intrusions, des menaces de réseau et des ensembles de données

Vu l'apparition de nouvelles méthodes des systèmes de détection d'intrusions, un ensemble de chercheurs ont pensé d'étudier quelques manuscrits pour donner un aperçu des internes des IDS et de la manière dont ils sont censés fonctionner. Cette étude est réalisée par Hanan Hindy et al [33] en 2018. L'étude est faite en analysant une dizaine d'articles revenant où principales recherches de l'IDS de la dernière décennie (2008-2018). Ces articles étudiés sont mentionnés dans des tableaux, où l'année est cité, le nom des auteurs de chaque article avec le nom de ce dernier en précisant les

données sur lesquelles le manuscrit se base et le nom des algorithmes utilisées pour la détection des attaques et l'attaque détectée. La référence de chacun de ces articles est citée dans le manuscrit [33]. Ce travail aboutit aux trois résultats que nous allons citer dans ce qui suit. L'étude a démontré que les recherches actuelles ne couvrent qu'environ 25 % des menaces. Les attaques les plus visées sont les attaques disponibles dans l'ensemble des données KDD-99 met en évidence par l'IDS. Le taux de détection d'un IDS en ligne se dégrade souvent lorsqu'il est utilisé sur un réseau occupé. La mise en oeuvre d'un système hybride peut être plus efficace où les hôtes jouent le rôle des capteurs pour le renseignement sur les attaques.

Il est aussi remarquable que l'apprentissage machine est utilisé par 97,25 % des IDSs étudiés. L'IDS basé sur l'apprentissage machine se concentre sur la détection d'intrusions dans les réseaux. Les algorithmes les plus utilisés sont ANN(Artificial Neural Network), k-means et SVM(Support Vector Machine). Les résultats atteints sont remarquables mais obtenus par des datasets obsolètes qui ne reflètent pas la réalité des attaques de réseau. La plupart des ensembles de données n'ont pas de véritables propriétés de vie ce qui dégrade l'efficacité des IDSs et peut refléter des fausses alertes ou manqué les nouvelles attaques qui n'existe pas encore dans le dataset utilisé et ils ne peuvent pas s'adapter aux changements constants des réseaux.

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

En 2018, Glass-Vanderlan et al [31] ont fait une étude sur les IDSs qui exploitent les sources des données de l'hôte pour détecter les attaques sur le réseau d'entreprises. Les articles étudiés montrent des modèles testés sur des ensembles des données de réseau, mais applicables aux IDSs basés sur l'hôte (HIDS). Selon l'article [31], 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 des 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 IDSs. 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 IDSs 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.

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 IDSs. 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 [31], 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 vu 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.

Plusieurs exemples d'étude sont présentés dans cet article [31] concernant les points cités et d'autres points sur les IDSs du registre Windows et les IDSs des systèmes de fichiers. Une sélection d'oeuvres décrit des tests d'algorithmes d'apprentissage machine standard qui sont applicables aux HIDS, mais qui relèvent des applications NIDS. Des recherches sont recommandées sur l'avancement et le développement dynamique des IDSs afin d'optimiser la sécurité.

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

Plusieurs études ont été réalisées afin d'améliorer la corrélation des alertes en se basant sur plusieurs critères. Dans cette section, nous allons aborder l'un des plus récent modèle réalisé qui peut y être intégré avec les IDS qui fonctionnent en ligne dans l'ordre de calcul de O(n). Ehsan Mahdavi et al (2019) [35] ont proposé un nouveau modèle de corrélation d'alertes qui consiste à minimiser le nombre des alertes détectées et éliminer les faux positifs. Ce modèle est nommé RACC(Real-time Alert Correlation method based on Code-books).

Le RACC permet une corrélation d'alertes en temps réel, en utilisant une matrice pour chaque scénario appelé code-book avec un index unique de chacune généré dans la phase hors ligne.

Les hypers alertes représentent les caractéristiques du dataset qui ne sont pas gardées dans la mémoire, mais plutôt dans le code-book, ce qui réduit le temps de corrélation des alertes. Ces caractéristiques sont extraites en utilisant les méthodes séquentielles qui tentent notamment de modéliser les relations de cause à effet entre les incidents. Ces derniers font partie des entrées XML qui est la structure utilisée dans ce modèle. L'idéal pour ces méthodes est de pouvoir prédire les prochains mouvements des utilisateurs malveillants en analysant les incidents actuels. Chaque ligne de la matrice doit représenter une alerte typique qui pourrait être utilisée dans plusieurs scénarios prévus et les colonnes de la matrice représentent un système protégé. Un élément de la matrice est représenté par un entier et n'occupe qu'un bit d'espace mémoire. Chaque scénario est placé dans un vecteur, est chaque vecteur contient la position de l'hyper alerte dans ce vecteur pour former un vecteur indice. Pour trouver ces indices qui permettent d'avoir le bon chemin d'accès au code-book, un arbre rouge-noir(red-black trees) est utilisé. Les noeuds de cet arbre prétendent les hypers alertes qui se pointent vers un vecteur indice. Le texte de l'hyper-alerte est la clé de tri de l'arbre Rouge-Noir.

2.4.1 Fonctionnement du modèle RACC

Dans cette partie, nous décrirons le fonctionnement du modèle RACC en manipulant la figure 2.1 qui détaille le schéma général de ce dernier.

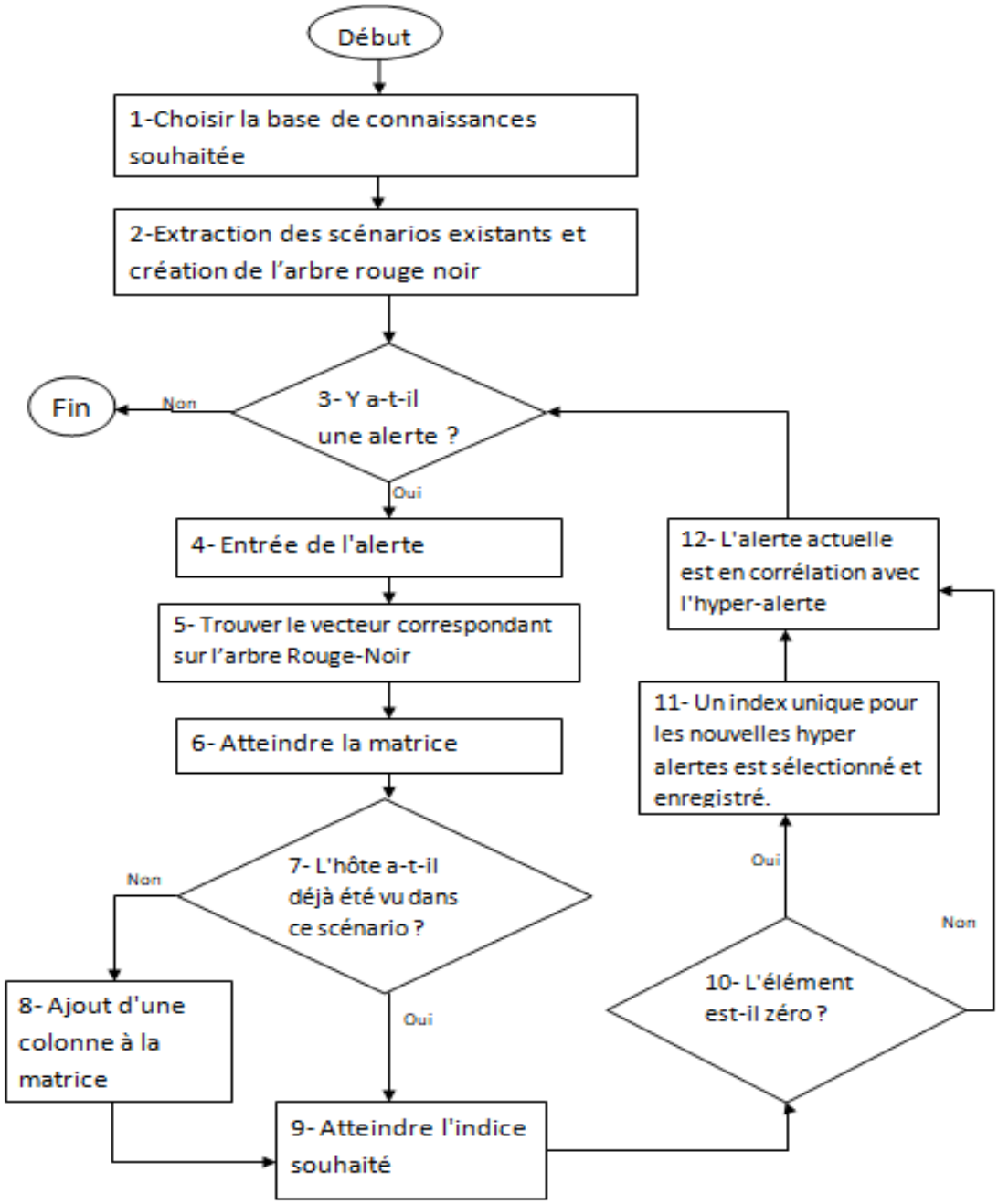

FIGURE 2.1 – Aperçu de la méthode RACC [35].

Premièrement, les auteurs ont choisi une base de données avec laquelle l'extraction des scénarios existants est faite pour former le code-book et articuler l'arbre Rouge-Noir avec tous les vecteurs. Ensuite, le programme récursif commence en vérifiant l'existence d'une alerte signalée. Si c'est le cas, un moteur de recherche sur l'arbre se lance pour trouver le vecteur d'indice qui va préciser la ligne correspondante à l'alerte signalée et dans quelle matrice. Les opérations matricielles ne sont pas liées au nombre de problèmes de sécurité et dans la phase en ligne, elles apparaîtront

comme des valeurs constantes. Un accès au scénario aura lieu. Après avoir détecté la ligne, une vérification des colonnes s'effectue. S'il y a une colonne dans la matrice relative à l'hôte dans lequel l'alerte est signalée, les éléments de cette colonne sont vérifiés, sinon une colonne pour ce dernier doit être ajoutée. L'intersection entre la ligne et la colonne repérée, représente l'élément souhaité. Si l'élément égal à zéro, ça veut dire qu'aucun signal n'a été déjà dénoncé sur cet hôte donc il est nécessaire de créer une méta-alerte et un index unique comme élément correspondant dans le code-book. Si l'élément est différent de zéro, la méta-alerte correspondante existe déjà et elle sera corrélée avec l'hyper-alerte qui lui convient. Ensuite, le programme se répète à l'existence de nouvelles alertes jusqu'à qu'il se termine.

2.4.2 Résultats de la corrélation d'alertes de la méthode RACC

Le modèle RACC a été mis en oeuvre en utilisant le langage C++ dans une distribution Ubuntu de l'environnement Linux. Il a été testé pour deux systèmes de détection d'intrusions qui sont les IDSs SNORT et RealSecure avec leurs bases de connaissances correspondant aux alertes. Ces bases de connaissances sont utilisées dans la mise en oeuvre de cette méthode en construisant les code-books pour trois ensembles de datasets. Les trois datasets sont TH qui contient des alertes générées par Snort IDS pour le trafic réel d'une société privée qui est fourni pour cette recherche et deux scénarios de type DARPA2000, où le premier scénario comporte 5 étapes et son objectif final est de mener une attaque DDoS et l'autre scénario utilise des requêtes de type HINFO envoyées à un fournisseur DNS2 au lieu de les envoyer à tous les hôtes comme dans le scénario précédent.

En faisant la comparaison avec d'autres méthodes qui utilisent des graphes de traitements nécessitent des parcours de graphes lourds, l'utilisation de la structure des code-books permet de réaliser des tâches similaires, plus simples et plus rapides ce qui illustre les avantages du RACC. Une autre comparaison aux autres modèles est effectuée pour examiner la validité de l'allégation de grande vitesse en termes de temps CPU. Les résultats de la figure qui suit sont calculés en utilisant un matériel similaire dans les mêmes conditions d'évaluation et en temps réel. La figure 2.2 désigne une courbe indiquant le temps CPU nécessaire pour traiter les alertes dans des épisodes de 30, 60, 90 et 120 minutes entre quatre méthodes inclue la méthode RACC. Il est clair que le RACC peut traiter les alertes et corréler les scénarios, en un temps nettement inférieur à celui des méthodes présentées dans [40], [45] et [27]. Les tests réalisés dans cet article [35] ont montré une augmentation considérable des performances et une diminution de la consommation de ressources par rapport aux autres méthodes de corrélation d'alertes en temps réel.

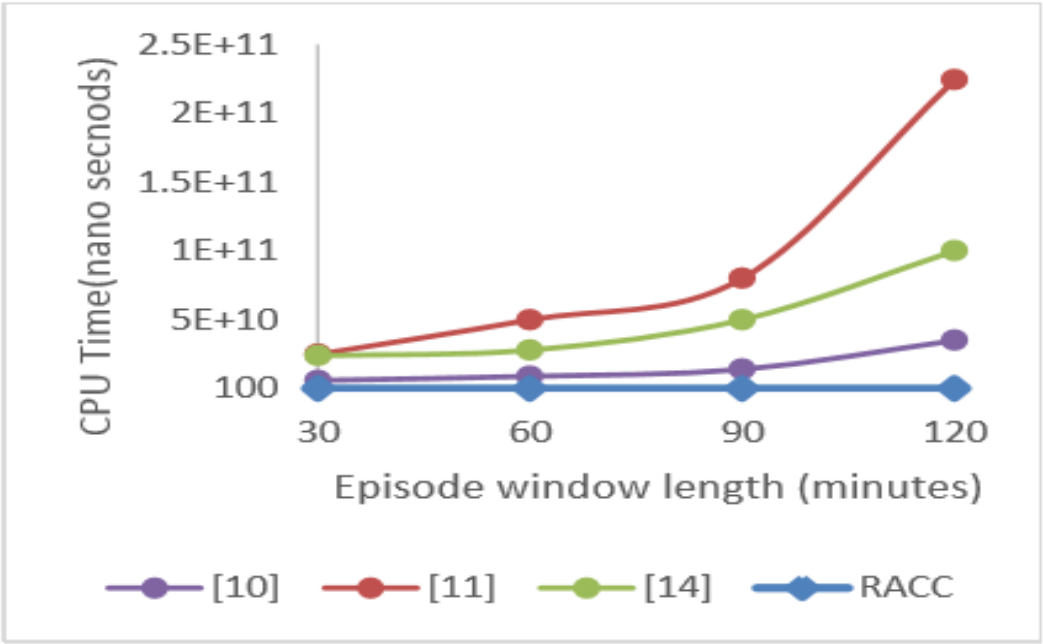

FIGURE 2.2 – Comparaison du temps CPU avec un matériel similaire dans le traitement des alertes Realsecure pour le premier scénario de DARPA2000 entre RACC et [40], [45] et [27], acquis à partir de [35].

2.4.3 Évaluation de la méthode RACC

La première préoccupation concernant la solution proposée, c'est l'augmentation du nombre de scénarios et de leur durée moyenne qui pourra avoir un impact direct sur la vitesse d'exécution et les ressources nécessaires vu que la précision de la méthode proposée dépend fortement de ses scénarios obtenus d'une source externe. Cette augmentation peut être entraînée à cause d'une redondance de communication entre les alertes. En outre, ça peut causer une vaste consommation de mémoire. L'utilisation de techniques d'apprentissage automatique pour extraire des scénarios peut étendre la méthode et réduire les coûts tout en améliorant la précision. Il est à noter aussi que cette étude néglige l'effet de corrélation des alertes dans la réduction des faux positifs. Il est possible de réduire les méta-alertes finales en imposant des limites sur le nombre d'étapes des scénarios pour diminuer les risques de cette négligence.

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

Une menace avancée persistante, également appelée APT (Advanced Persistent Threats), écrite par Ibrahim Ghafir et al [28], est une cyberattaque la plus inquiétée à l'échelle mondiale car elle entraîne généralement des vols de données et des pertes pour la société concernée. L'attaque

de l'APT est une attaque persistante elle se déroule en plusieurs étapes, son but principal est l'espionnage puis l'exfiltration des données. Du coup, APT est estimé comme une nouvelle version plus complexe. D'après une longue recherche, les auteurs ont constaté que l'APT représente un défi pour les méthodes de détections (pare feux.) car elle peut agir sans être détectée pendant des mois voir des années et elle utilise des vulnérabilités inconnues contrairement aux autres.

2.5.1 Architecture de la MLAPT

Dans cet article [28], les auteurs ont développé un nouveau système basé sur l'apprentissage machine, appelé MLAPT(Machine-Learning Advanced Persistent Threats), qui peut détecter et prédire avec précision et rapidité les attaques APT de manière holistique. Le MLAPT passe par trois phases principales : la détection des menaces, la corrélation des alertes et la prévision des attaques, comme il est illustré dans la figure 2.3.

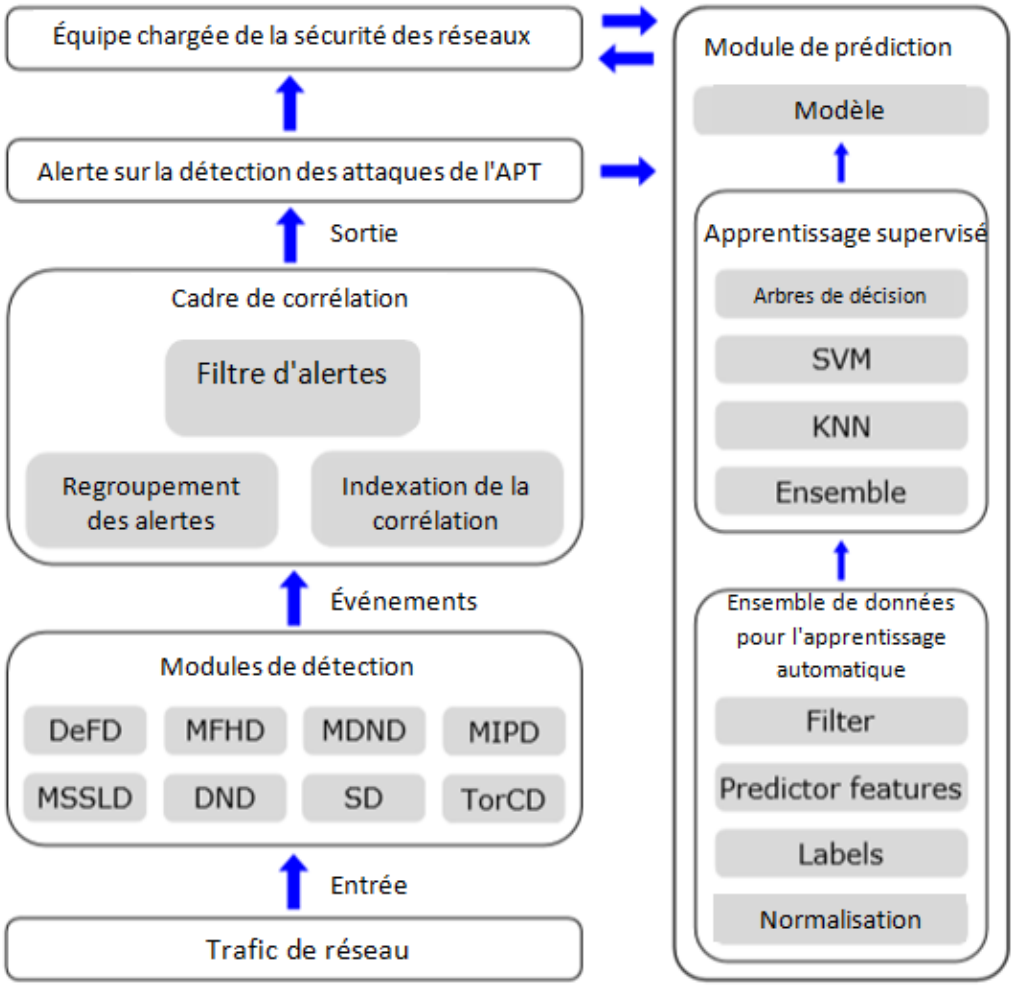

FIGURE 2.3 – L'architecture de la MLAPT [28].

Dans un premier temps, le trafic du réseau est scanner et traité pour détecter les éventuelles techniques utilisées dans le cycle de vie de l'APT. À cette fin, huit modules de détection ont été développés ; chaque module met en oeuvre une méthode pour détecter une technique utilisée dans l'une des étapes de l'attaque APT et il est indépendant des autres modules. Le MLAPT a mis en oeuvre huit modules pour détecter les techniques les plus couramment utilisées dans le cycle de vie de l'APT. Les résultats de cette phase sont des alertes, également appelées événements, déclenchés par des modules individuels. Les alertes déclenchées par les différents modules de détection sont ensuite transmises au cadre de corrélation. L'objectif du cadre de corrélation est de trouver des alertes qui pourraient être liées et appartenir à un scénario d'attaque APT. Le processus de cette phase comporte trois étapes principales : Le filtrage des alertes pour identifier les alertes redondantes ou répétées, 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 Présentation de quelques travaux sur la détection d'intrusions et le Machine Learning les alertes de chaque groupe.

Dans la phase finale, un module de prédiction basé sur l'apprentissage machine est utilisé par l'équipe de sécurité du réseau pour déterminer la probabilité des premières alertes pour développer une attaque APT complète. Cela permet à l'équipe de sécurité du réseau de prédire l'attaque APT dans ses premières étapes et d'appliquer la procédure requise pour l'arrêter avant qu'elle ne soit terminée et minimiser les dégâts. La détection de l'APT est différente de la prédiction. La détection peut avoir lieu 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.

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

D'après l'analyse des performances des quatre systèmes de détection APT existants, une comparaison est faite entre le système développé MLAPT et ces systèmes actuels.

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

FIGURE 2.4 – Une comparaison entre le MLAPT et d'autres systèmes existants [28].

Le système TerminAPTor, un détecteur APT [22] utilise le suivi des flux d'informations pour trouver les liens entre les alertes élémentaires est considéré comme le système le plus efficace en termes de taux positifs réel avec un TPR (taux de faux positifs) de 100 %. Les développeurs ont noté que le système TerminAPTor a un taux élevé de faux positifs, ce dernier a besoin que les alertes soient fournies par d'autres systèmes et ne peut pas fonctionner de manière indépendante [28]. Par contre, il est remarquable que le système CC [47], a un taux de faux positifs plus faible (0%) et ne permet pas la détection en temps réel et ne dépend que de la détection d'une seule étape du cycle de vie de l'APT [28]. Ainsi que le système basé sur le spear phishing [23], il a un TPR de 97,2 %, le FPR (taux de faux négatifs) de 14,2 % est considérablement élevé. Le système basé sur le contexte [30], a un TFP significativement élevé de 27,88 % alors que le TFP n'a pas été fourni par les auteurs [28]. Réaliser un taux élevé de vrais positifs signifie que le taux de faux positifs augmente également. Par suite, l'équilibre entre le TPR et le FPR est une exigence essentielle pour tout système de détection. D'après les résultats demandés, un équilibre remarquable entre les valeurs de TPR et FPR, ce qui donne la possibilité à MLAPT de générer des événements et avoir un fonctionnement autonome.

Ces événements couvrent quatre étapes de détection de l'APT ce qui réduit les faux positifs et donne plus de possibilités de détection de l'APT au cas où l'une des étapes serait manquée. En outre ce système aide à minimiser les dommages et de prévenir d'autres effractions [28].

D'après les auteurs, le MLAPT est le seul système qui peut prévoir l'APT dans ses premières étapes avec une précision de prévision de 84,8 %, ce qui empêche l'attaquant d'atteindre l'objectif de l'ex filtration des données.

2.6 Apprentissage de la machine contradictoire pour les filtres antispams

Les IDSs ont connu une place importante dans tout système informatique vu son rôle pour la sécurité de ces derniers. On a vu dans les articles précédent que les IDSs permettent de détecter des menaces en se basant sur les caractéristiques des anciennes attaques déjà connue, corréler des alertes pour une meilleure gestion et une bonne précision de détection. Ces attaques peuvent être sous forme de logiciel malveillant, des téléchargements furtifs, des attaques par déni de service ou bien cassage de mot de passe, etc. Et Depuis l'apparition des courriels, une autre attaque a vu sa naissance, c'est l'envoi des messages non sollicité qui peuvent rapporter des risques de sécurité et un danger à la vie privée du destinataire, ces messages sont appelés spam. L'utilisation des techniques du Machine Learning a retracé des méthodes remarquables et plus efficaces pour la détection des spams. Dans cet article écrit par B. Kuchipudi et al en 2020 [34], nous allons voir une de ces méthodes les plus récemment utiliser qui se base sur la classification. Présentation de quelques travaux sur la détection d'intrusions et le Machine Learning.

2.6.1 Fonctionement du modèle

Trois techniques contradictoires sont étudiées dans l'article [34] pour la mise en oeuvre du filtre de messages de spam. Ces techniques enrayées le filtre et le rend inutile ce qui réduit le taux de prédiction et considère les messages comme ham (bénin). La première technique, c'est l'injection des mots ham ou bien le remplacement des abréviations par le mot complet dans des messages sans changer le sens de la phrase, c'est-à-dire enrichir le message par des mots anodin malgré qu'il contienne des mots spam pour qu'il soit classé comme bénin après le filtrage. La deuxième technique, c'est la construction de nouveaux messages à partir d'un seul message en cherchant pour chaque mot du message un ensemble de synonyme. Ensuite, les messages construits peuvent être classés comme spam ou ham, cela peut mettre le modèle en conflit entre la bonne et la mauvaise décision vu la forte homologie entre le contenu des messages appartenant aux déférentes classes. Au final, un autre problème peut causer de fausses réponses du filtre, vu que la prédiction se base sur le nombre d'apparitions des mots. Les attaquants utilisent les espaces entre les lettres de chaque mot pour qu'il ne soit pas prédit comme spam alors que cela ne change pas les risques de ce mot qu'il peut provoquer.

On prenant en considération les techniques citées, les auteurs de cet article [34] ont pu réaliser une méthode réussie pour une meilleure prédiction en utilisant un modèle baysien qui est connue comme l'un des plus puissants modèles de classification pour des problèmes a plusieurs caractéristiques. Les données du dataset sont codées en supprimant les mots d'arrêt et la ponctuation. Plusieurs mots ayant une signification similaire sont liés ou regroupés dans le processus de transformation. Ensuite, la transformation obtenue sera convertie en une matrice de comptage de fréquence sur laquelle le modèle baysien prédit le résultat.

2.6.2 Résultat du prédiction

Le dataset utilisé pour tester le modèle contient 5 572 messages où 747 sont des spams. La transformation citée dans le paragraphe précédent est appliquée sur ce dataset puis il est séparé en deux parties de données, données du test et données du pratique, où 1 115 messages sont pris pour le test. La performance du modèle est évaluée par deux mesures principales, la précision et le rappel. Plus le chiffre approche de 1, plus le modèle sera plus efficace. La figure 2.5 montre le score de ces deux derniers.

	Precision	Recall	F1-score	support
ham	0.96	1.00	0.98	961
spam	1.00	0.73	0.84	154
micro avg	0.96	0.96	0.96	1115
macro avg	0.98	0.86	0.91	1115
weighted avg	0.96	0.96	0.96	1115

FIGURE 2.5 – La précision du modèle Naive Bayes [34].

	Predicted Spam	Predicted Ham
Actual Spam	970	0
Actual Ham	34	111

FIGURE 2.6 – Présentation des résultat par la matrice de confusion [34].

Le résultat de prédiction peut être présenté sous forme d'une matrice de confusion, figure 2.6, qui permet d'exprimer des indicateurs de performance où on peut voir le nombre de messages prédit et calculer l'erreur de prédiction. On remarque que le taux de faux négatifs est égale à 0 ce qui veut dire qu'aucun message bénin ni détecter comme message spam alors que 34 messages spams détectés comme bénins représentant le nombre de faux positifs. La question posée dans ce résultat, c'est : comment va-t-il réagir le modèle si un mot inexistant dans la base de données apparaît ? C'est la réponse demandée pour tout modèle basant sur des anciennes données.

2.7 Conclusion

Dans ce chapitre, nous avons abordé quelques méthodes de détection avec une brève explication de leur fonctionnement et nous avons entamé d'autres articles qui indiquent une énorme utilisation des algorithmes d'apprentissage dans le domaine de détection d'intrusions. Selon les résultats obtenus et les études effectuées, ces algorithmes ont prouvé leur efficacité et peuvent être un point de base irremplaçable pour le réglage de ce genre de problèmes. Malgré cet avancement remarquable, une infinité d'études pour minimiser ces intrusions sont exigées. Tant que les intrus essaient sans hésitation d'accéder aux informations illégales ou personnelles, d'autres intrusions apparaissent. L'avancement des réseaux sociaux peut mettre la vie privée de l'utilisateur en danger ce qui exige des moyennes plus fiables de sécurité. Les modèles de détection des spam basés sur le Machine Learning sont de plus en plus demandés et utilisés par tout appareil vu leur importance pour une communication sécurisée.

Chapitre 3

Proposition d'un filtre anti-pourriel

3.1 Introduction

Dans ce chapitre, nous proposons notre modèle de filtrage des spams basé sur le traitement automatique du langage naturel (NLP, Natural Language Processing). Ce modèle est testé sur quatre algorithmes d'apprentissage. Nous commençons par présenter quelques concepts de base liés à notre proposition, puis nous présenterons notre proposition et les étapes d'implémentation. Pour enfin, nous synthétisons et évaluons les résultats obtenus.

3.2 Concepts de base liés à notre proposition

Dans cette section, nous allons définir quelque concepts de base liés à notre proposition.

3.2.1 Spam

Le terme "spam" fait généralement référence à des courriers électroniques commerciaux non sollicités qui sont généralement envoyés en masse à des milliers, voire des millions de destinataires [44]. Le spam peut être classé en fonction de l'objectif du spammeur. De nombreux spammeurs envoient leur courrier électronique en masse pour des raisons publicitaires, par exemple, ils envoient des publicités commerciales ou participent à des campagnes politiques, tandis que d'autres ont en tête une sorte de fraude criminelle ou distribuent des logiciels malveillants, tels que des virus ou des chevaux de Troie [43]. C'est à partir de l'année 2004 que les spammeurs donnent aux auteurs de virus les moyens de distribuer leurs marchandises et maintenant les spammeurs peuvent faire plus que simplement envoyer du courrier indésirable, ils peuvent contrôler les ordinateurs de leurs victimes [44]. Les messages légitimes, par opposition aux spams sont souvent désignés par le mot ham, probablement pour dire que " spam, c'est mauvais, mais ham, c'est bon " [25].

3.2.2 Filtre antispam

Pour faire face aux pertes massives résultant du spam, beaucoup de recherches ont été effectuées pour trouver des solutions plus efficaces à ce problème qui touche chaque utilisateur d'un courrier électronique. En se basant sur plusieurs critères d'un message spam, des filtres antispam ont été créés. Le rôle d'un filtre antispam, comme son nom l'indique, est de filtrer les messages d'entrée, pour ensuite faire traverser les messages bénins et dégager les messages spams. Plusieurs approches sont utilisées pour filtrer le spam. Souvent, si un message en cours d'examen vérifie un certain nombre de critères, donc le message est refusé. En général, une combinaison des critères est nécessaire pour atteindre un niveau d'efficacité satisfaisant, mais elle n'est pas forcément triviale à trouver. Soit les critères sont renseignés explicitement tel que l'approche naïve, soit on se contente de définir une heuristique permettant à l'algorithme de construire lui-même cet ensemble de critères tout comme dans l'approche alternative, l'apprentissage artificiel. Dans l'apprentissage artificiel, un ensemble d'exemples est utilisé avec les étiquettes associées, de taille suffisante pour être représentatif de l'ensemble des messages et de laisser un certain algorithme apprendre la relation fonctionnelle pouvant exister entre l'ensemble d'exemples et l'ensemble de classes. L'objectif est que la relation, apprise sur un petit nombre d'exemples, puisse être généralisée à l'ensemble des messages possibles, avec un faible taux d'erreur [25]. L'approche d'apprentissage automatique la plus connue dans le filtrage des spams est celle du classificateur Bayes naïfs, elle calcule et utilise la probabilité de certains mots / expressions apparaissant dans les exemples les plus connus (messages) afin de classer de nouveaux exemples (messages). NaïveBayes a été montré pour être très bien réussi à catégoriser les documents texte. Filtres bayé-siens (méthode statistique) analysent les mots du message à l'intérieur d'un e-mail pour calculer la probabilité que un message est un spam ou non. Le calcul basé sur des mots qui déterminent que le message est un spam et les mots qui déterminent que le message n'est pas du spam [29].

3.2.3 Mesures d'estimation de performance

Un modèle de prédiction peut être satisfaisant, mais n'est pas parfait. Le choix d'une méthode compétente de filtrage du spam nécessite une évaluation du rendement et une comparaison du meilleur résultat généré. Pour faire, quelques mesures d'estimation de performance ont été établies. Parmi ces mesures, on note les suivantes :

3.2.3.1 Matrice de confusion

Table NxN qui résume la réussite des prédictions d'un modèle de classification, c'est-à-dire la corrélation entre les étiquettes et les classificat ions du modèle. L'un des axes d'une matrice de confusion est l'étiquette prédite par le modèle et l'autre l'étiquette réelle. N correspond au nombre de classes. Dans un problème de classification binaire, N=2 [3]. La figure 3.1 montre une vue de la matrice de confusion théorique NxN.

		Classification					
Référence		Classe 1	Classe 2	Classe 3	Classe 4	Total	Erreur d'omission
	Classe 1	a	...	...	...	A'	1 - (a / A')
	Classe 2	...	b	...	...	B'	1 - (b / B')
	Classe 3	...	...	c	...	C'	1 - (c / C')
	Classe 4	...	...	...	d	D'	1 - (d / D')
	Total	A	B	C	D	N	
	Erreur de commission	1 - (a / A)	1 - (b / B)	1 - (c / C)	1 - (d / D)		

FIGURE 3.1 – Matrice de confusion théorique [17].

Les matrices de confusion contiennent suffisamment d'informations pour calculer diverses statistiques de performance, notamment la précision et le rappel [3]. Les quatre mesures essentielles sont VP, FP, VN et FN. VP représentent le nombre de valeurs positives, VN le nombre de valeurs négatives prédites correctement, FP représente le nombre de valeurs positives mal estimé et FN le nombre mal estimé de valeurs négatives.

3.2.3.2 Précision

La précision est une donnée qui est prise sur la base d'un manque d'information. Dans la classification binaire, la précision peut être rendue égale à des valeurs prédictives positives. La formule suivante est la règle de précision [39] :

$$Precision = (\frac{TP}{TP + FP}) * 100\%$$

3.2.3.3 Rappel

Le rappel, connu sous le nom de sensibilité dans la classification binaire, concerne les données de suppression qui ont été récupérées avec succès à partir de données pertinentes pour la requête. Il est défini par l'équation suivante [39] :

$$Rappel = (\frac{TP}{TP + FN}) * 100\%$$

3.2.3.4 Accuracy

Accuracy , ou bien l'exactitude est un pourcentage du total des données qui est identifié et évalué. La règle d'exactitude est la suivante [39] :

$$Accuracy = (\frac{TP + TN}{TP + TN + FP + FN}) * 100\%$$

3.2.4 Notre proposition

Après avoir détaillé quelques méthodes de détection des intrusions dans le chapitre précédent, nous présentons dans ce chapitre un modèle de détection des spams qui apporte des solutions à certaines limites observées en utilisant les bases du Machine Learning. Notre approche consiste à analyser un ensemble de messages partagés en deux classes, spam qui représente des messages malveillants et une classe ham qui représente des messages bénins. Le but de cette analyse est de réaliser un modèle qui fournit une bonne décision de classification afin de minimiser le taux d'erreur pour une communication plus rassurante. Pour répondre à la question introduite dans le chapitre précédent, nous avons intégré dans ce chapitre une méthode qui peut être l'une des solutions pour le problème proposé. Comme tous les modèles supervisés, l'apprentissage ce fait en se basant sur des anciennes données, donc si un mot non reconnu déjà par le modèle est mis à la pratique, la performance de prédiction diminue et de fausses résultats seront estimés. Pour cela, nous avons pensé à la relation qui réunit chaque mot à un autre, c'est l'alphabet. Chaque mot est une combinaison d'un ensemble de lettres ordonnées, sauvent répétées, formant une unité de sens. Cette répétition de lettre diffère d'un message bénin à un message spam, d'où, nous avons exploité ces deux méthodes (nombre de répétitions de chaque lettre et le nombre d'apparitions de chaque mot dans un message) et nous avons fait une combinaison entre les deux pour former un modèle avec une performance remarquable de 98% de précision.

3.2.5 Architecture de notre proposition

La figure 3.2 décrit l'architecture qui définit les étapes de réalisation du modèle.

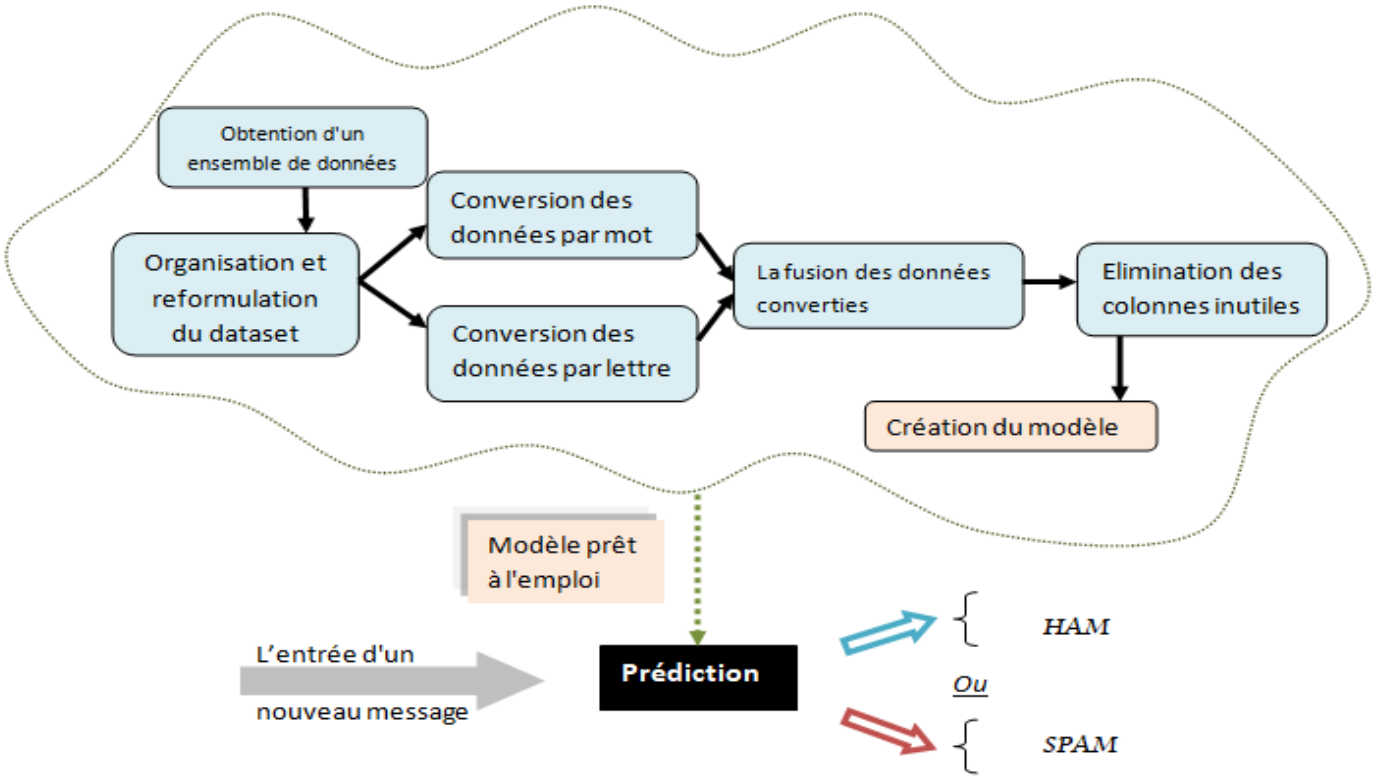

FIGURE 3.2 – L'architecture du modèle proposé.

La phase d'apprentissage encerclé dans la figure 3.2 comport les différentes étapes de l'implémentation d notre modèle avant d'être employé. L'enchaînement d'exécution de ces étapes est représenté par des flèches. Après la création du modèle, on passe à la prédiction. À chaque entrée d'un nouveau message, un filtrage de ce dernier sera effectué et l'un des résultats possibles sera prédit, soit le message est un spam, soit il est bénin.

3.3 Détails d'implémentation

En se basant sur les recherches précédentes, nous avons réussi à réaliser un modèle qui permet de filtrer les messages spams. Les outils et environnement de programmation utilisés pour la réalisation de l'étude proposée ainsi les bibliothèques essentielles pour l'apprentissage automatique en Python sont définis dans l'annexe A . Dans cette section, nous allons mettre en clair les étapes de l'implémentation montré dans l'architecture du modèle proposé.

3.3.1 Obtention d'un ensemble de données

Nous utilisons un ensemble de données de spam publiquement disponible sur Kaggle [4] (le même dataset utilisé dans l'article "Adversarial Machine Learning for Spam Filters" [34] étudié dans le chapite 2 section 2.6. Le dataset contient un total de 5 572 messages, dont 747 sont étiquetés comme des spams et 4 825 qui reste sont des messages ham. Le dataset est sous forme d'un fichier (.csv : Comma-Separated Values) qui est un fichier Excel. Comme illustre la figure 3.3, il est formé par cinq colonnes dont lesquelles on utilise que les deux premières. La première colonne indique si le message est un spam ou ham et la deuxième colonne contient l'ensemble des messages.

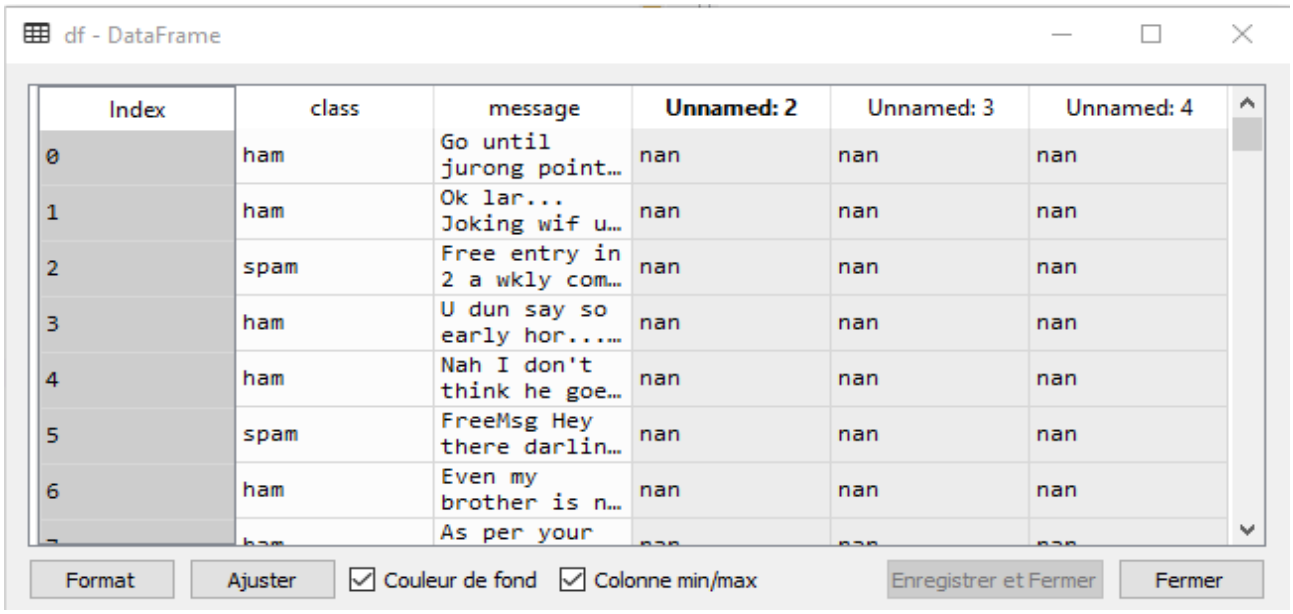

FIGURE 3.3 – L'ensemble de données de spam à traiter, publiquement disponible sur Kaggle [4].

3.3.2 Organisation et reformulation du dataset

Une mauvaise organisation du dataset invoque l'anarchie des données et exploite plus d'espace mémoire avec une détermination critiquable. L'organisation comprend la suppression des trois dernières colonnes inutiles pour ensuite, appliquer une reformulation des deux colonnes qui restent. Pour cela, nous avons appliqué une fonction qui permet de mettre tous les messages en minuscule et éliminer les ponctuations, les virgules, les points d'exclamation, etc afin d'avoir que des lettres en minuscule dans chaque mot du message entier comme nous montre la figure 3.4. Ainsi, une suppression des lignes répétées est recommandée vu que c'est le même message qu'elle représente. Une autre reformulation de la colonne classe est nécessaire pour l'exécution de quelques commandes où, les lignes pourtant le terme Ham sont remplacés par 0 et les lignes spam se transforment en 1.

FIGURE 3.4 – L'ensemble de données après l'organisation et la reformulation.

3.3.3 Conversion des données

Au cours de cette étape, nous avons exercé deux rubriques de conversion sur le même jeu de données que nous avons réalisé dans l'étape précédente. L'une des rubriques compte le nombre d'apparitions de chaque lettre dans un message, l'autre conversion réalise le même travail en comptant le nombre de chaque mot. Chaque mot ou lettre est représenté par colonne et chaque ligne contient les nombres calculés. Une fusion des deux résultats est effectuée pour avoir un dataset complet sur lequel on va lancer le modèle.

3.3.4 Élimination des colonnes inutiles

L'insertion des espaces dans un message spam est l'une des méthodes utilisées pour contredire la prédiction. Vu que l'espace est un caractère donc il est considéré comme une lettre d'où une colonne pour ce dernier a été générée l'heure de la conversion. Après quelques tests effectués sur le modèle, l'effacement de la colonne espace ajoute plus de performance à la prédiction vu que sa présence dégrade la précision. Le dataset que nous avons formé englobe 7708 colonnes par 5169 lignes comme illustre la figure 3.5.

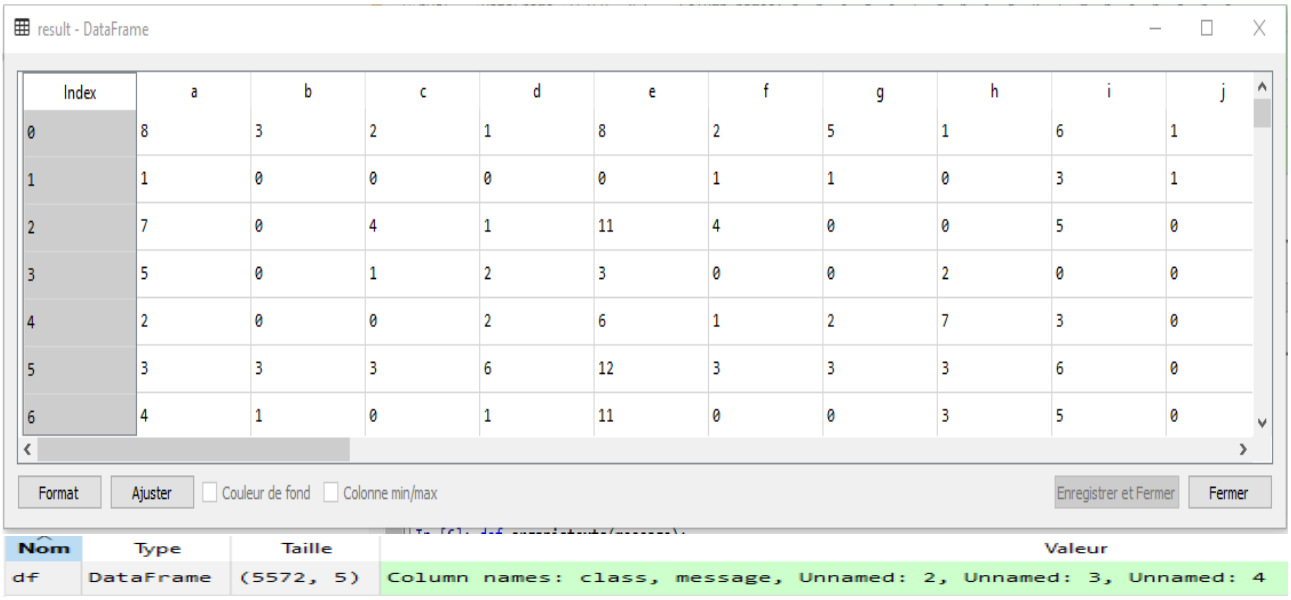

FIGURE 3.5 – L'ensemble des données obtenues.

3.3.5 Création du modèle

Après avoir effectué les opérations précédentes, nous sommes arrivés à former un ensemble des données prêtes à la manipulation. Le choix du modèle dépend des données fournies. Un ensemble d'algorithmes d'apprentissage sont stockés dans une bibliothèque dans l'environnement scientifique spyder nommé sklearn. Sans être explicitement programmés, un modèle peut être utilisé en lui accordant les données d'entrées. Dans notre cas, nous entraînons un modèle d'apprentissage supervisé. La mise en pratique consacre 20 % de tous les messages pour le test et le reste des données en tant qu'un ensemble de formation. Cette résolution permet de distinguer quel algorithme est plus idéal pour résoudre la tâche en pratique et passer à la prédiction de nouveaux messages.

3.4 Synthèse des résultats

Maintenant que nous avons défini les données nécessaires pour la réalisation et expliqué l'architecture formant le modèle proposé, nous pouvons commencer à générer les résultats d'exécution. Nous avons testé quatre différents algorithmes de classification et réalisé une comparaison globale sur la meilleure précision standard et la mesure de rappel généré. Ça nous permet d'évaluer les modèles de classification et de trouver le plus satisfaisant du point de vue de la détection des spams.

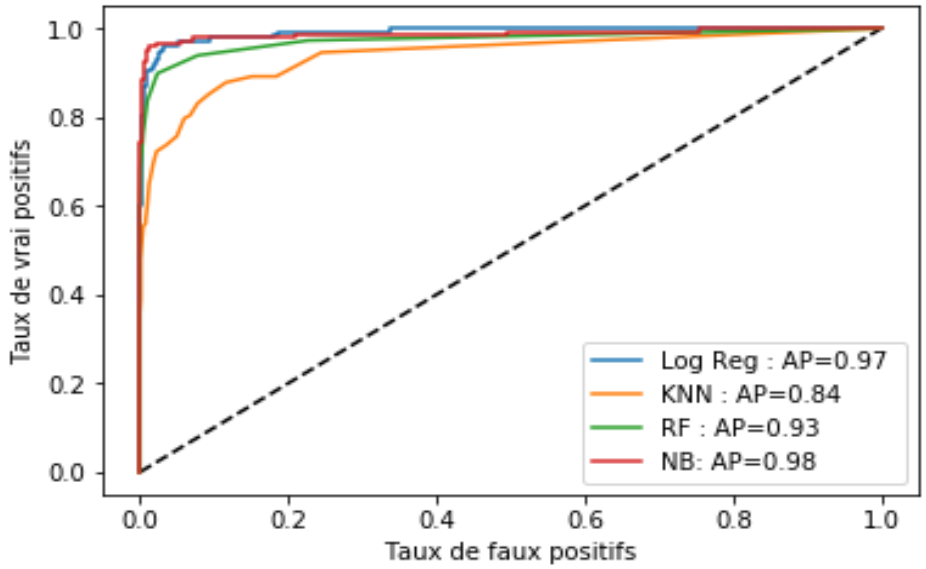

FIGURE 3.6 – Courbe ROC(Receiver Operating Characteristic).

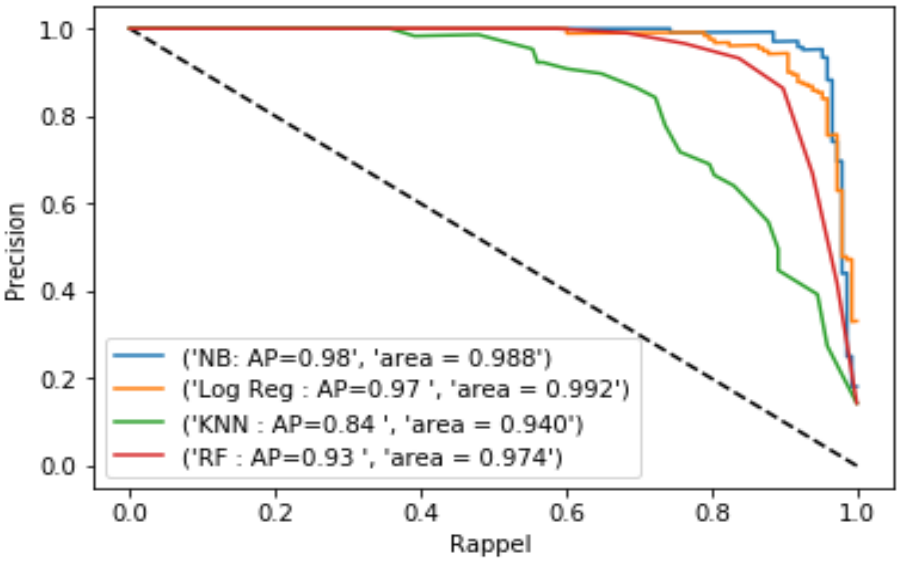

FIGURE 3.7 – Courbe Precision vs Recall.

Les deux courbe illustré dans la figure 3.6 et 3.7 nous permettent de visualiser les probabilités pour les mesures, taux de faux positifs en fonction du vrai positifs et précision vs rappel. On conclu que le programme de classification naïve bayes est plus performant que les autres en prenant en considération AP(Average precision) et ROC(Receiver Operating Characteristic) area.

Pour plus de détails, la figure 3.8 décrit la matrice de confusion qui prend en compte les deux types d'erreurs possibles : les faux positifs et les faux négatifs ainsi retrace les résultats de l'évaluation des performances du filtre anti-pourriel naïve bayes en termes de précision. On note une

```
...: print(classification_report(Y_test,y_pred))
...: print(accuracy_score(Y_test,y_pred))
...: print(fbeta_score(Y_test,y_pred,beta=0.5))
...:
...: confusion_matrix(Y_test,clf.predict(X_test))
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       886
           1       0.95      0.95      0.95       148

   micro avg       0.99      0.99      0.99      1034
   macro avg       0.97      0.97      0.97      1034
weighted avg       0.99      0.99      0.99      1034

0.9854932301740812
0.9510869565217392
Out[11]:
array([[879,   7],
       [  8, 140]], dtype=int64)
```

FIGURE 3.8 – Résultat de l'évaluation des performances du filtre anti-pourriel naïve bayes proposé et la matrice de confusion.

valeur de justesse égale à 0,985, soit 98 %, ce qui conclu que notre classificateur affiche des résultats très satisfaisants du point de vue de la détection des messages malveillants.

3.5 Évaluation et validation du modèle

Cette étape nous permet de tester le modèle sur des nouvelles données d'entrée. La figure 3.9 nous montre le code de texte qui s'exécute en arrière plan de la page web montré dans la figure 3.10. Nous avons créé cette page sous forme d'application web. Dans la fenêtre à gauche, on peut saisir quelques messages puis en cliquant sur le bouton filtré, le résultat de prédiction s'affiche dans la fenêtre à droite.

```
184 ##############################################
185 ##############################test new msg#######################
186 text=['l']
187 vect=cv.transform(text).toarray()
188 vect2=ngram_vectorizer.transform(text).toarray()
189 #fr = vect2 + vect
190 #vect2.extend(vect)
191 #rslt = pd.concat(fr, axis=1, sort=False)
192 arr = np.concatenate((vect2, vect),axis=1)
193 fr=np.delete(arr, 0, 1)
194 pr=clf.predict(fr)
195 #fr.shape()
196 pr
197 #
198 #
```

FIGURE 3.9 – Partie du code : test sur de nouvelle données.

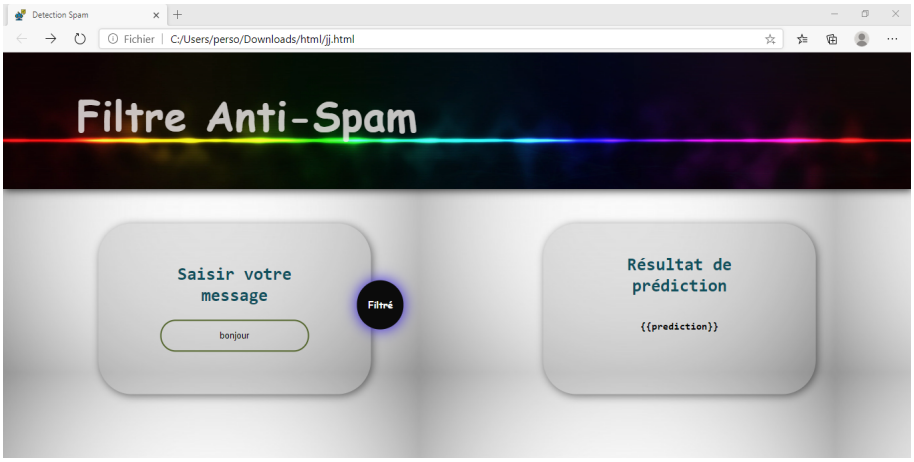

FIGURE 3.10 – Interface web du modèle proposé.

3.6 Conclusion

Dans ce chapitre, nous avons présenté l'architecture de notre système de filtrage des spams ainsi une vue complète sur ce système. Nous avons étudié les résultats obtenus dans les différents algorithmes Log Reg, RF, NB et KNN avec des différentes représentations où nous avons pris l'algorithme NB (naïf bayes) comme meilleur classificateur grâce à ces bons résultats générés. Notre proposition apporte une nouvelle méthode de classification des messages spams. Dans les anciennes méthodes, on passe d'une implémentation scalaire, ensemble de messages dans la colonne message, à une implémentation vectorielle ou chaque mot est stocké par colonne indiquant son apparition par ligne. Mais cela ni pas suffisant pour de nouvelles prédictions d'où, nous avons implémenté une deuxième relation qui réunit chaque mot ou message à un autre. Cette méthode ni pas limitée seulement sur les mots déjà existants dans le dataset, mais aussi basée sur chaque apparition d'une lettre dans un message ce qui fait même si le modèle ni pas familiarisé avec un mot du message d'entrée, il sera filtré en fonction des lettres qu'il compose. Nous avons conclu le chapitre par une illustration du déploiement de notre modèle sur une page web pour une bonne visualisation du fonctionnement.

Conclusion et perspectives

Dans ce mémoire, nous avons présenté, dans le chapitre 1, deux principaux axes. D'abord, les systèmes de détection d'intrusions où nous avons décrit les types qui construisent ce système ainsi l'architecture fonctionnelle et la classification de ses éléments. Par la suite, nous avons expliqué les méthodes d'apprentissages Machine Learning.

Dans le chapitre 2, nous avons accordé entre les deux concepts « Machine Learning » et « les IDSs » en illustrant avec un cas d'étude sur la détection des spams malveillants où, nous avons traité plusieurs articles sur les IDSs et finaliser par le problème des spams qui touche presque tous les utilisateurs du courrier électronique.

Dans le chapitre 3, nous avons proposé une nouvelle technique de modélisation et d'analyse des attaques par des moyennes de communication numérique. Cette méthode permet de filtrer les courriels électroniques à base d'un classificateur bayésien multinomial. Le résultat de prédiction porte une valeur de précision remarquable 98 % ce qui prouve sa performance.

Comme perspectives nous envisageons un chiffrement des données pour un déploiement sécurisé pour des utilisations externes et nous comptons faire une étude sur les message contenant des images.

Annexe A

Outils et environnement de réalisation

Les outils et environnement de programmation utilisés pour la réalisation de l'étude proposée :

Python

langage de programmation polyvalent, qui existe déjà depuis assez longtemps, Quido van Rossum , son créateur, ayant débuté son développement en 1990. Stable et mature, il s'agit d'un langage de très haut niveau, dynamique et orienté objet [36]. Il soutient de multiples paradigmes de programmation au-delà de la programmation orientée objet, comme la programmation procédurale et fonctionnelle [5]. Un fichier python porte l'extension (.py).

Anaconda

gestionnaire de paquets, un gestionnaire d'environnement, une distribution de données scientifiques Python/R et une collection de plus de 7 500 paquets open-source. Anaconda est gratuit et facile à installer, et il offre un soutien communautaire gratuit [6].

Spyder

un environnement scientifique puissant écrit en Python, pour Python. Il est conçu par et pour les scientifiques, les ingénieurs et les analystes de données. Il présente une combinaison unique de fonctionnalités avancées d'édition, d'analyse, de débogage et de profilage d'un outil de développement complet avec les capacités d'exploration de données, d'exécution interactive, d'inspection approfondie et de visualisation d'un ensemble scientifique [7].

Jupyter Notebook

application Web à source ouverte qui vous permet de créer et de partager des documents contenant du code en direct, des équations, des visualisations et du texte narratif. Les utilisations incluent : nettoyage et transformation de données, simulation numérique, modélisation statistique, visualisation de données, apprentissage automatique, etc . Le projet Jupyter est un projet open-

source à but non lucratif, né du projet IPython en 2014. Il a été évolué pour soutenir la science des données interactives et le calcul scientifique dans tous les langages de programmation. Jupyter sera toujours un logiciel 100% open-source, libre d'utilisation pour tous [8].

HTML (HyperText MarkupLanguage)

HyperText MarkupLanguage, en français "langage de balisage d'hypertexte", est un langage informatique de balises qui permet de crée des pages web sous forme d'un ensemble de document ou pages reliées entre elles par des liens hypertextes [9].

CSS (Cascading Style Sheets)

Cascading Style Sheets, en français "feuilles de style en cascade" est un langage informatique permettant de mettre en forme le contenu d'un site et de gérer le design d'une page HTML [9].

Bibliothèques essentielles pour l'apprentissage automatique en Python

bibliothèque de programmes ou bien librairie logicielle est un ensemble de fonctions utilitaires, regroupées et mises à disposition afin de pouvoir être utilisées sans avoir à les réécrire. Les fonctions sont regroupées de par leur appartenance à un même domaine conceptuel (mathématique, graphique, tris, etc) [10]. La bibliothèque standard de Python est très grande, elle offre un large éventail d'outils [5].

Dans ce qui suit, nous allons définir les bibliothèques utilisées dans notre implémentation :

Pandas

Bibliothèque open source, sous licence BSD (Berkeley Software Distribution), qui fournit des structures de données et des outils d'analyse de données performants et faciles à utiliser pour le langage de programmation Python [11].

Matplotlib

Bibliothèque complète pour la création de visualisations statiques, animées et interactives en Python [12].

Seaborn

Bibliothèque de visualisation de données en Python basée sur matplotlib. Elle fournit une interface de haut niveau pour dessiner des graphiques statistiques attrayants et informatifs [13].

NumPy

Paquet fondamental pour le calcul scientifique en Python. C'est une bibliothèque Python qui fournit un objet de tableau multidimensionnel, divers objets dérivés (tels que des tableaux et des matrices masqués), et un assortiment de routines pour des opérations rapides sur des tableaux, y compris des opérations mathématiques, logiques, de manipulation de formes, de tri, de sélection, d'entrées/sorties, de transformées de fourier discrètes, d'algèbre linéaire de base, d'opérations statistiques de base, de simulation aléatoire et bien plus encore [14].

sklearn

Un module Python intégrant des algorithmes classiques d'apprentissage machine dans le monde étroitement lié des paquets scientifiques Python (numpy, scipy, matplotlib) [15].

Pickle

Le module pickle met en œuvre des protocoles binaires pour sérialiser et désérialiser une structure d'objet Python. Le "pickling" est le processus par lequel une hiérarchie d'objets Python est convertie en un flux d'octets, et le "unpickling" est l'opération inverse [5].

Le module Re

Re , C'est un module en python qui fournit des opérations sur les expressions rationnelles [5].

NLTK (Natural Language Toolkit)

Plate-forme de pointe pour la construction de programmes Python destinés à fonctionner avec des données en langage humain. Parmi les choses simples que peuvent être faite avec NLTK, on peut citer : marquer du texte, identifier des entités nommées et afficher un arbre d'analyse [20].

Références Bibliographiques

[16] M. R. Amini. *Apprentissage machine de la théorie à la pratique*. Eyrolles, 2015.

[17] N. Baghdadi, C. Mallet et M. Zribi. *QGIS et applications en agriculture et forêt*. ISTE Editions, 2018.

[18] S. Balech et C. Benavent. *Les techniques du NLP pour la recherche en sciences de gestion*. 2019.

[19] K. Belkhatmi et O. Benamara. *Mise en place d'un système de détection et de prévention d'intrusion*. mémoire de master 2, Université A/Mira de Béjaïa, 2016.

[20] S. Bird et E. Loper. *Natural Language Processing with Python : analyzing text with the natural language toolkit*. "O'Reilly Media, Inc", 2009.

[21] A. Boulaiche. *Modèle hybride basé sur des données qualitatives et quantitatives pour la détection d'intrusions*. thèse de doctorat. Université A/Mira de Béjaïa, 2018.

[22] G. Brogi et V. V. T. Tong. *Terminaptor : Highlighting advanced persistent threats through information flow tracking, in : New Technologies, Mobility and Security (NTMS)*. 8th IFIP International Conference on, IEEE, pp. 1-5, 2016.

[23] J. V. Chandra, N. Challa et S. K. Pasupuleti. *A practical approach to e-mail spam filters to protect data from advanced persistent threat, in : Circuit, Power and Computing Technologies (ICCPCT)*. International Conference on, IEEE, pp. 1-5, 2016.

[24] M. Rémi Coulom. *Apprentissage par renforcement utilisant des réseaux de neurones, avec des applications au contrôle moteur*. Thèse du doctorat de l'institut national polytechnique de GRENOBLE, 2002.

[25] J. M. M. da Cruz. *Spam : classement statistique de messages électroniques : Une approche pragmatique*. presses des MINES, collection mathematique et informatique, paris, 2012.

[26] H. Debar, B. Morin et F. Cuppens. *Détection d'intrusions : corrélation d'alertes*. Sécurité informatique RSTI - TSI, pages 359 à 390, 2004.

[27] H. Farhadi, M. AmirHaeri et M. Khansari. *Alert Correlation and Prediction Using Data Mining and HMM*. The ISC International, Journal of Information Security, vol. 3, no. 2, pp.77-101, 2011.

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

[29] C-E. Gherabi. *Mémoire présenté pour l'obtention du diplôme de Master Académique de M'SILA*. UNIVERSITE MOHAMED BOUDIAF - M'SILA, Algérie, 2017.

[30] P. Giura et W. Wang. *A context-based detection framework for advanced persistent threats, in : Cyber Security (CyberSecurity)*. International Conference on, IEEE, pp. 69-74, 2012.

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

[32] L. Hamza. *Génération automatique de scénario d'attaques pour les systèmes de détection d'intrusion*. Mémoire de magistère Université Abderrahmene Mira de Béjaia, Algérie, 2005.

[33] H. Hindy, D. Brosset et E.n Bayne. *A Taxonomy and Survey of Intrusion Detection System Design Techniques*. Network Threats et Datasets, 2018.

[34] B. Kuchipudi, R. T Nannapaneni et Q. Liao. *Adversarial Machine Learning for Spam Filters*. In Proceedings of the 15th International Conference on Availabality et Security, pp. 1-6, 2020.

[35] E. Mahdavi, A. Fanian et Fatima Amini. *A real-time Alert Correlation Method Based on Code-books for Intrusion Detection Systems*. Computers Security, 2019.

[36] Alex Martelli. *python en concentré*. edition O'Reilly, paris, 2004.

[37] G. Fantozzi N. Godin P. Reynaud. *Émission acoustique et durabilité des composites*. ISTE Editions Ltd, 2018.

[38] Jean-Marc Quéré. *WinDev 9 : Implémentation de méthodes décisionnelles,* Editions ENI, 2005.

[39] F. Rahmad, Y. Suryanto et K. Ramli. *Performance Comparison of Anti-Spam Technology Using Confusion Matrix Classification*. Series : Materials Science et Engineering (Vol.879,No. 1,p.012076). IOP Publishing, 2020.

[40] A. A. Ramaki, M. Amini et R. E. Atani. *RTECA : Real time episode correlation algorithm for multi-step attack scenarios detection*. computers security, vol. 49, pp. 206-219, 2015.

[41] S. Russell et P. Norvig. *Intelligence artificielle : Avec plus de 500 exercices*. Pearson Education, 2010.

[42] G. Saint-Cirgue. *apprendre le machine learning en une semaine*. machinelearnia.com, 2019.

[43] G. Schryen. *Anti-Spam Measures Analysis and Design*. Springer-Verlag Berlin Heidelberg, 2007.

[44] P. H. Gregory M. Simon et Spyware For Dummies. *Blocking Spam Spyware For Dummies*. John Wiley Sons, 2005.

[45] 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.

[46] C. Della Vedova. *Introduction à la régression logistique*. 2020.

[47] X. Wang, K. Zheng et X. Niu. *Detection of command and control in advanced persistent threat based on independent access,in : Communications (ICC)*. International Conference on, IEEE, pp. 1–6., 2016.

Références Webliographiques

[1] https ://www.groupe-hli.com/machine-learning-dans-industrie, Consulté le 27/03/2020.

[2] https ://www.moniaetmoi.com/machine-learning, Consulté le 27/03/2020.

[3] https ://developers.google.com/machine-learning, Consulté le 13/10/2020.

[4] https ://www.kaggle.com/ uciml/sms-spam-collection-dataset , Consulté le 18/09/2020.

[5] https ://docs.python.org/, Consulté le 29/09/2020.

[6] https ://docs.anaconda.com/anaconda/, Consulté le 29/09/2020.

[7] https ://docs.spyder-ide.org/current/index.html, Consulté le 29/09/2020.

[8] https ://jupyter.org/about, Consulté le 29/09/2020.

[9] https ://glossaire.infowebmaster.fr, Consulté le 22/10/2020.

[10] https ://www.techno-science.net/definition/1470.html, Consulté le 29/09/2020.

[11] https ://pandas.pydata.org/docs/, Consulté le 03/10/2020.

[12] https ://matplotlib.org/, Consulté le 03/10/2020.

[13] https ://seaborn.pydata.org/, Consulté le 03/10/2020.

[14] https ://numpy.org/doc/stable/user/whatisnumpy.html, Consulté le 03/10/2020.

[15] https ://www.kite.com/python/docs/sklearn, Consulté le 03/10/2020.

RÉSUMÉ

L'utilisation des systèmes de détection d'intrusions est l'un des moyens d'offrir un environnement sécurisé et rassurant pour les Utilisateurs des systèmes informatiques. Des mises à jours et amélioration pour ces systèmes de protection sont recommandés vue l'apparition régulière des nouvelles vulnérabilités. Les recherches ont montrées le rôle important du Machine Learning dans la construction et la réalisation de nouvelles techniques plus satisfaisante et qui peuvent prédire les nouvelles attaques plus rapidement et efficacement pour préparer la contre-mesure la mieux adaptée. Dans notre cas, nous avons traité le problème des spams. Les courriers électroniques sont devenus indispensables de notre quotidien mais qui importe des inconvénients que seule une bonne détection peut la régler. Nous avons crié un modèle de filtrage des spams à base de NLP, qui prédit l'origine d'un message d'entrée avec une precision de 98 %. Grâce à l'utilisation du Machine Learning, des solutions efficaces peuvent être réalisé en vue de renforcer la capacité de détection des systèmes de détection d'intrusions.

mot clé : Machine Learning, Sécurité Informatique, IDSs, Spam, Filtre Antispam.

ABSTRACT

The use of intrusion detection systems is one of the means to offer a secure and reassuring environment for the Users of computer systems. Updates and enhancements to these protection systems are recommended as new vulnerabilities regularly appear. Research has shown the important role of Machine Learning in the construction and realization of new techniques that are more satisfactory and that can predict new attacks more quickly and efficiently in order to prepare the most appropriate countermeasure. In our case, we dealt with the problem of spam. E-mails have become an indispensable part of our daily lives, but they have disadvantages that only a good detection can solve. We cried out a spam filtering model based on NLP, which predicts the origin of an incoming message with an accuracy of 98 %. Through the use of Machine Learning, effective solutions can be realized to enhance the detection capability of intrusion detection systems.

key words : Machine Learning, Computer Security, IDSs, Spam, Antispam filter.