REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLAB DE BLIDA 1

FACULTE DES SCIENCES

DEPARTEMENT D'INFORMATIQUE

PROJET DE FIN D'ETUDE POUR L'OBTENTION DU DIPLOME DE MASTER EN SECURITE DES SYSTEMES D'INFORMATION

Mémoire réalisé par :Douid Rania

THEME :

Système de détection d'intrusion réseau basé sur L'algorithme de Classification KNN

Organisme d'accueil : Sonatrach

Promotrice : Mme N. Boustia **Encadreur : Mr T. Boudheb**

Année Universitaire : 2018-2019

L'avant propos:

Je remercie tout d'abord Dieu tout puissant de m'avoir donné le courage, la force et la patience d'achever ce modeste travail.

Je tiens avant tout à remercier Mme Boustia pour SA patience et sans qui la rédaction de ce mémoire n'aurait simplement pas eu lieu. A mon encadreur qui a su me guider tout au long de ce projet. Un grand merci à Demiai Ahmed pour la patience dont il a su faire preuve à mon égard durant son aide, ainsi que pour ses nombreux conseils.

Enfin, les mots les plus simples étant les plus forts, j'adresse toute mon affection à ma famille, et en particulier à mes parents, leur confiance, leur tendresse, leur amour me portent et me guident tous les jours. Merci pour avoir fait de moi ce que je suis aujourd'hui. Je vous aime. Pour terminer, un grand merci aux personnes qui ont cru en moi et qui m'ont permis d'arriver au bout de ce projet.

Résumé :

La cybercriminalité augmente de plus en plus ces dernières années ce qui pousse les administrateurs à sécuriser convenablement leurs réseaux informatiques. Ils ont recours à la détection d intrusions pour détecter les attaques à l'entrée du réseau ou déceler les intrusions passées inaperçues à travers les pare-feux et les routeurs filtrants.
Pour cela, deux approches sont actuellement utilisées : l'approche comportementale et l'approche par scénarios. Nous utiliserons l'approche par scenarios avec les algorithmes de fouilles de données appliqués sur les données fournies par le projet NSL KDD CUP. Nous procédons, ensuite, à une analyse des résultats qui nous permet de situer les méthodes l'une par rapport à l'autre. Nous nous sommes principalement investis dans l'étude du KNN ainsi que la méthode bayésienne, les vecteurs machines (svm), la régression linéaire et leurs variantes.
Mots-clés : détections d'intrusions, algorithme, fouille de données, NSL KDD CUP, KNN, méthode bayésienne, les vecteurs machines (svm), la régression linéaire, détections.

Summary:

Cybercrime has been growing in recent years, prompting administrators to securely secure their computer networks. They use intrusion detection to detect attacks at the network entrance or to detect unnoticed intrusions through firewalls and filtering routers.

For this, two approaches are currently used: the behavioral approach and the scenario approach. We will use the scenario approach with the data mining algorithms applied to the data provided by the NSL KDD CUP project. We then proceed to an analysis of the results which allows us to situate the methods with respect to each other. We mainly invested in the KNN study as well as the Bayesian method? Machine vectors (svm) and their variants.

Keywords: intrusion detections, algorithm, data mining, NSL KDD CUP, KNN, Bayesian method, machine vectors (svm), detections.

Listes des acronymes et abréviation :

IDS	intrusions détection system.
SI	Système d'Information
DOS	Denial of Service
TCP	Transmission control Protocol
IP	Internet Protocol
U2R	User to Root
R2L	Remote to Local
ARP	Address Resolution Protocol
DNS	Domain Name Server
XSS	Cross Site Scripting
HTML	Hypertext Markup Language
ACL	access Control List
DMZ	Demilitarized Zone
VPN	Virtual Private Network
KNN	K Nearest Neighbors
SVM	Support Vector Machine
TN	True Negative
FN	False Negative
FP	False Positive
TP	True Positiv

Sommaire

Liste des figures :

Liste des tableaux :

Liste des équations :

Introduction générale

Dans la période actuelle où Internet croît à un rythme exponentiel, la sécurité informatique est devenue un problème critique.

L'exploitation de fausses alarmes IDS n'est pas un nouveau domaine de recherche. Manganaris et al. ont analysé les flux d'alarmes pour trouver les règles d'association. Ils ont suggéré un cadre de traitement des alarmes qui filtre les fausses alarmes pour le centre d'opérations réseau d'IBM, qui fournit aux clients des services de détection d'intrusion en temps réel. Leur approche consiste à caractériser le comportement normal de l'IDS en termes de groupes d'alarmes fréquents avec un minimum d'occurrences dans des rafales d'alarmes. Ensuite, les alarmes entrantes ont été classées pour rechercher des ensembles d'alarmes non fréquentes considérées comme suspectes.

Julisch, également d'IBM, a proposé de rechercher des groupes d'alarmes et des formes généralisées de fausses alarmes afin d'identifier les causes premières. Il a observé que 90% des fausses alarmes correspondaient à un petit nombre de causes premières. Connaissant les causes profondes, l'expertise humaine peut ajuster les IDS régulièrement ou supprimer les causes profondes afin de réduire le nombre de fausses alarmes de 82%, comme le montrent les expériences de Julisch. Il a également exploité des alarmes IDS pour les règles d'épisode, qui tentent de prédire l'ensemble des alarmes qui s'ensuit lorsqu'un ensemble spécifique d'alarmes se produit. Il a estimé que les règles sont utiles car, avec la connaissance de tels modèles d'alarmes représentant des utilisations légitimes des systèmes protégés, des alarmes très similaires (supposées légitimes également) peuvent être filtrées facilement à l'avenir.

Cependant, dans son rapport, les règles relatives aux épisodes n'offraient que 1% du taux de réduction des alarmes et 99% des alarmes restaient réservées au traitement manuel.

Récemment, l'exploration de données et l'apprentissage automatique ont fait l'objet de nombreuses recherches sur la détection des intrusions dans le but d'améliorer la précision du classifieur de détection. Le jeu de données NSL KDD est le jeu de données le plus utilisé dans le domaine de la recherche. La sélection de caractéristiques, importantes sur la base d'une approche de sélection de caractéristiques basée sur un ensemble approximatif, a conduit à une simplification du problème et à des taux de détection plus rapides et plus précis.

Les technologies d’exploration de données ont montré leur capacité à réduire de plus de la moitié les fausses alarmes. Notre approche décrite dans le présent projet réduit davantage le taux de fausses alarmes à l’aide du classificateur KNN.

Le mémoire est structuré comme suit :

Le premier chapitre sera 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 sera une présentation et explication globale du machine Learning ainsi que les algorithmes que nous allons utiliser dans notre projet.

Dans le troisième chapitre, nous passons à la conception et la mise en œuvre de notre IDS.

Et enfin le quantième chapitre sera consacré a la partie résultat et perspective.

Chapitre 1 : Système de détection d'intrusion réseau

1. Introduction :

L'internet a facilité le flux d'informations, personnelles, financières et autres. En même temps, il a également promu autant de dangers. Les utilisateurs malveillants recherchent des proies vulnérables comme les systèmes sans correctifs, les systèmes affectés par des chevaux de Troie et les réseaux exécutant des services peu sûrs. Des alertes sont nécessaires pour prévenir les administrateurs et les membres de l'équipe de sécurité qu'une effraction s'est produite afin qu'ils puissent répondre en temps réel au danger. Les systèmes de détection d'intrusions ont été conçus pour jouer le rôle d'un tel système d'alerte.

Dans ce chapitre nous présentons tout d'abord la notion de système de détection d'intrusions ainsi que son architecture. Nous présentons également la classification des IDS, dans ce cadre plusieurs critères sont pris en compte .Nous commençons par la classification selon la méthode d'analyse qui découpe les IDS en deux approches (comportementale et par signatures), enfin nous allons mettre le point sur la détection d'intrusions réseau.

2. Définition :

Un système de détection d'intrusions (IDS, de l'anglais Intrusion Détection System) est un périphérique ou processus actif qui analyse l'activité du système et du réseau pour détecter toute entrée non autorisée et / ou toute activité malveillante. La manière dont un IDS détecte des anomalies peut beaucoup varier ; cependant, l'objectif principal de tout IDS est de prendre sur le fait les auteurs avant qu'ils ne puissent vraiment endommager vos ressources. [26]

Les IDS protègent un système contre les attaques, les mauvaises utilisations et les compromis. Ils peuvent également surveiller l'activité du réseau, analyser les configurations du système et du réseau contre toute vulnérabilité, analyser l'intégrité de données et bien plus... [27]

Les IDS traditionnellement suivent deux critères :

- Fiabilité : toute intrusion doit effectivement donner lieu à une alerte. Une intrusion non signalée constitue une défaillance de l'IDS, appelée faux négatif.
- Pertinence des alertes : toute alerte doit correspondre à une intrusion effective. Toute « fausse alerte » (appelée également faux positif) diminue la pertinence de l'IDS. Un IDS est

parfaitement fiable en absence de faux négatif ; il est parfaitement pertinent en l'absence de faux positif. [28]

Limitations:

- il est tout simplement active, pas proactive (optimiste).
- Il ne peut pas empêcher l'attaque.
- il n'est pas automatisé, il a besoin d'importantes ressources humaines pour leur gestion.
- il ne peut pas offrir une protection complète pour les ressources. C'est juste une couche supplémentaire.
- il ne peut pas compenser les lacunes des protocoles réseau.
- il ne peut pas protéger tous les types d'attaques. Il a des limites.
- il ne peut pas résister à des volumes élevés et des vitesses élevées de trafic Internet. [29]

3. Caractéristiques d'un IDS :

Les caractéristiques suivantes sont souhaitables dans un IDS :

- Fonctionnement en permanence avec une supervision manuelle minimale.
- Etre tolérant aux pannes dans le sens où il doit récupérer après une défaillance ou une réinitialisation de la machine.
- Résister aux tentatives de corruption, c'est-à-dire, il doit pouvoir détecter s'il a subit lui-même une modification indésirable
- Utiliser un minimum de ressources du système sous surveillance
- Etre facilement configurable pour implémenter une politique de sécurité spécifique d'un réseau.
- S'adapter au cours du temps aux changements du système surveillé et du comportement des utilisateurs.
- Etre difficile à tromper.

Comme la taille des réseaux a tendance à croitre, on peut ajouter les caractéristiques suivantes:

- Etre scalable.
- Etre robuste, c'est-à-dire l'arrêt d'un composant ne doit pas entrainer une défaillance totale. [30]

4. Classification des IDS :

Nous allons présenter ici la technologie de détection d'intrusion d'une manière taxonomique. Il y a plusieurs types d'IDS disponibles aujourd'hui, caractérisé par différente approche de la surveillance et de l'analyse. Chaque approche a ses avantages et des inconvénients distincts. Toutes les approches peuvent être décrites en termes d'un modèle d'IDS. (Voir la figure 1)

- L'emplacement d'IDS
- Les méthodes de détection
- Les types de réponse
- Fréquence d'utilisation [31]

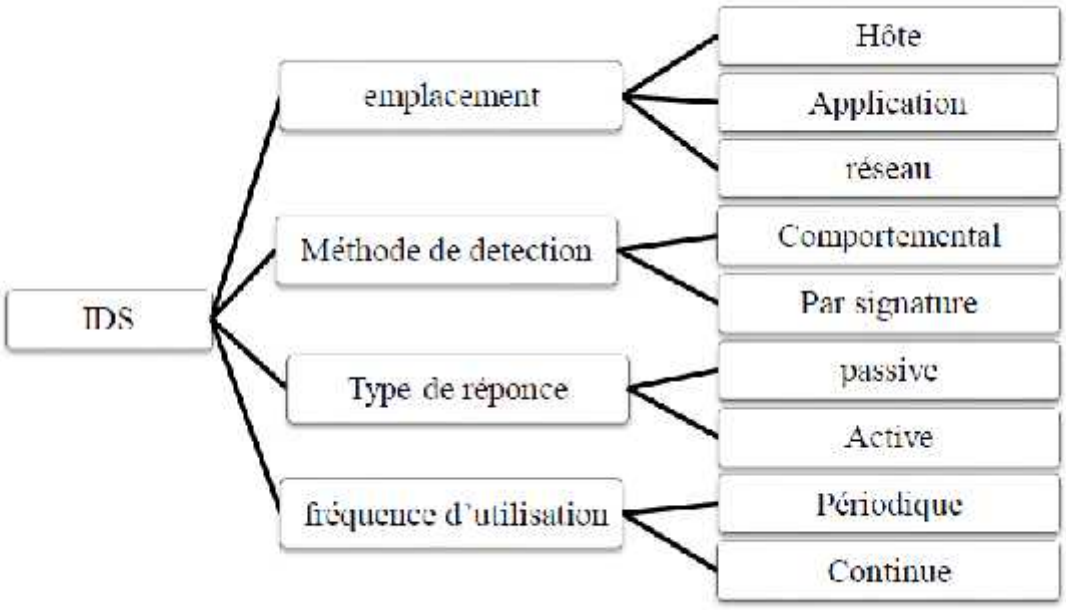


Figure 1 : Classification des IDS. [Online, 31]

4.1 Emplacement d'un IDS :

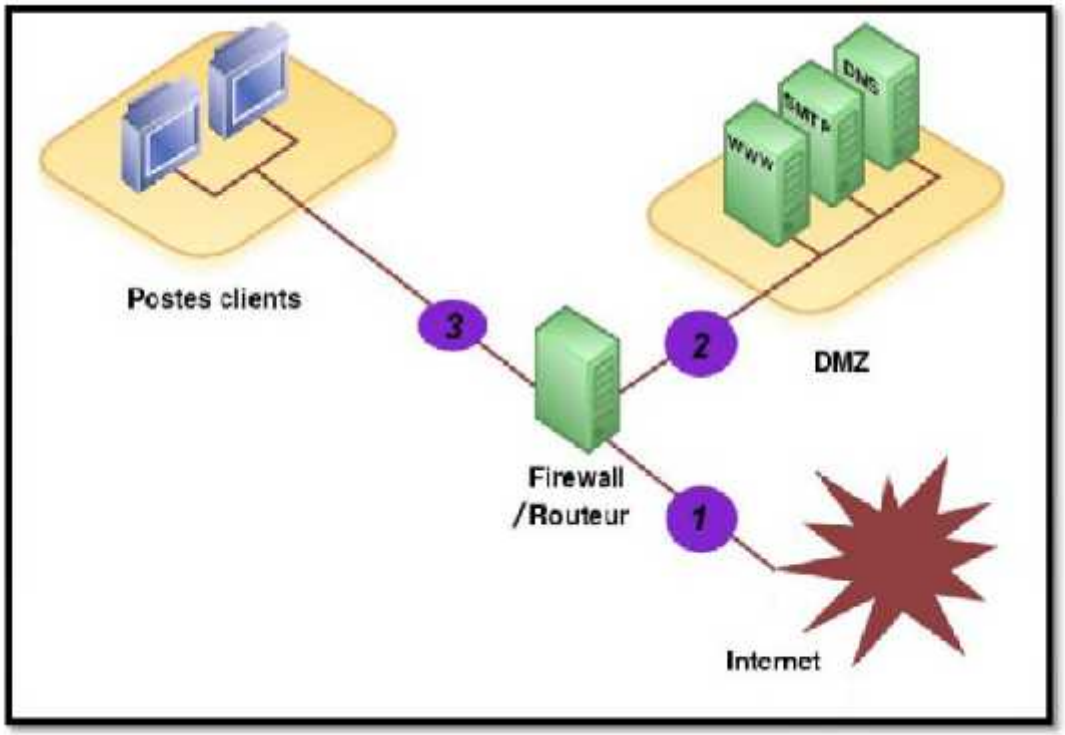


Figure 2 : Emplacement d'un IDS. [Online, 32]

a) Système de détection d'intrusion de type hôte (HIDS) :

Les systèmes de détection d'intrusion basés sur l'hôte ou HIDS (Host IDS) analysent exclusivement l'information concernant cet hôte. Comme ils n'ont pas à contrôler le trafic du réseau mais "seulement" les activités d'un hôte. Avec une grande fiabilité et précision, l'IDS peut déterminer exactement quels utilisateurs ou quels processus sont impliqués dans une attaque. Avec ces types de systèmes, le résultat peut être déterminée à la différence du NIDS, le HIDS peut accéder directement et de contrôler les fichiers de données et les processus habituellement visés par les attaques.

Ces IDS utilisent généralement les traces d'audit du système d'exploitation afin de fournir une information sur l'activité de la machine.il possède un certain nombre d'avantages : il est possible de constater immédiatement l'impact d'une attaque et donc de mieux réagir. Grâce à la quantité d'information étudiée, il est possible d'observer les activités se déroulant sur l'hôte avec précision et d'optimiser le système en fonction des activités observées. De plus, les HIDS sont extrêmement complémentaires des NIDS. En effet, ils permettent de détecter plus facilement les attaques de type "Cheval de Troie", alors que ce type d'attaque est difficilement

détectable par un NIDS. Les HIDS permettent également de détecter des attaques impossibles à détecter avec un NIDS, car elles font partie du trafic crypté.
Néanmoins, ce type d'IDS possède également ses faiblesses, qui proviennent de ses qualités : du fait de la grande quantité de données générées, ce type d'IDS est très sensible aux attaques de type Dos, qui peuvent faire exploser la taille des fichiers de logs. (Position 3 de la figure 2) [33]

b) Détection d'Intrusion basée sur une application :

Les IDS basés sur les applications (AIDS) sont un sous-groupe des IDS hôtes. Ils contrôlent l'interaction entre un utilisateur et un programme en ajoutant des fichiers de log afin de fournir de plus amples informations sur les activités d'une application particulière. Puisque vous opérez entre un utilisateur et un programme, il est facile de filtrer tout comportement notable. Un ABIDS (application based IDS) se situe au niveau de la communication entre un utilisateur et l'application surveillée. L'avantage de cet IDS est qu'il lui est possible de détecter et d'empêcher des commandes particulières dont l'utilisateur pourrait se servir avec le programme et de surveiller chaque transaction entre l'utilisateur et l'application. De plus, les données sont décodées dans un contexte connu, leur analyse est donc plus fine et précise.
Par contre, du fait que cet IDS n'agit pas au niveau du noyau, la sécurité assurée est plus faible, notamment en ce qui concerne les attaques de type "Cheval De Troie". De plus, les fichiers de log générés par ce type d'IDS sont des cibles faciles pour les attaquants et ne sont pas aussi sûrs, par exemple, que les traces d'audit du système. Ce type d'IDS est utile pour surveiller l'activité d'une application très sensible, mais son utilisation s'effectue en général en association avec un HIDS. Il faudra dans ce cas contrôler le taux d'utilisation CPU des IDS afin de ne pas Compromettre les performances de la machine. (Position 2 de la figure 2). [31]

c) Système de détection d'intrusion réseau (NIDS) :

Le rôle essentiel d'un IDS réseau est l'analyse et l'interprétation des paquets circulant sur ce réseau. L'implantation d'un NIDS sur un réseau se fait de la façon suivante : des capteurs Sont placés aux endroits stratégiques du réseau et génèrent des alertes s'ils détectent une attaque. (Position 1 de la figure 2). Ces alertes sont envoyées à une console sécurisée, qui les analyse et les traites éventuellement. Cette console est généralement située sur un réseau isolé, qui relie uniquement les capteurs et la console. .

Les avantages des NIDS sont les suivants : les capteurs peuvent être bien sécurisés Puisqu'ils se contentent d'observer le trafic et permettent donc une surveillance discrète du réseau, les attaques de type scans sont facilement détectées, et il est possible de filtrer le trafic.
Les NIDS sont très utilisés et remplissent un rôle indispensable, mais ils présentent Néanmoins de nombreuses faiblesses. En effet, la probabilité de faux négatifs (attaques non détectées comme telles) est élevée et il est difficile de contrôler le réseau entier. De plus, ils doivent principalement fonctionner de manière cryptée d'où une complication de l'analyse des paquets. Pour finir, à l'opposé des IDS basés sur l'hôte, ils ne voient pas les impacts d'une attaque. [34]

4.2 Méthode de détection :

Deux approches de détection ont été proposées :
L'approche comportementale modélise le comportement normal des utilisateurs, du système informatique et de l'activité réseau. Ensuite, toute déviation par rapport à la normale constitue un événement suspect. Tandis que l'approche par signature, elle, s'appuie sur un modèle constitué des sections interdites dans le système d'informatique, ce modèle s'appuie sur la connaissance des techniques employées par les attaquants : on tire des scénarios d'attaque et on recherche dans les traces d'audit leur éventuelle survenue. [34]

4.2.1 L'approche comportementale :

Les détecteurs d'intrusions comportementaux reposent sur la création d'un modèle de référence représentant le comportement de l'entité surveillée en situation de fonctionnement normal. Ce modèle est ensuite utilisé durant la phase de détection afin de pouvoir mettre en évidence d'éventuelles déviations comportementales. Pour cela, le comportement de l'entité surveillée est comparé à son modèle de référence. Une alerte est levée lorsqu'une déviation trop importante (notion de seuil) vis-à-vis de ce modèle de comportement normal est détectée. Le principe de cette approche est de considérer tout comportement n'appartenant pas au modèle de comportement normal comme une anomalie symptomatique d'une intrusion ou
D'une tentative d'intrusion.

L'approche comportementale possède un certain nombre d'avantages et d'inconvénients :

Les avantages :

- l'analyse comportementale n'exige pas des connaissances préalables sur les attaques.

- Elle permet la détection de la mauvaise utilisation des privilèges.
- Elle permet de produire des informations qui peuvent être employées pour définir des signatures pour l'analyse basée connaissance.

Les inconvénients :

- Les approches comportementales produisent un taux élevé des alarmes type faux positif en raison des comportements imprévisibles des utilisateurs et des réseaux.
- Ces approches nécessitent des phases d'apprentissage pour caractériser les profils de comportement normaux.
- Les alarmes génériques par cette approche ne sont pas significatives. [30]

4.2.2 L'Approche par scénario ou signature :

Dans cette approche, les détecteurs d'intrusion reposent sur la création d'une base de motifs qui représente des scénarios d'attaque connus au préalable et qui sera utilisé par la suite, le plus souvent en temps réel, sur les informations fournies par les sondes de détection. C'est donc un système de reconnaissance de motifs qui permet de mettre en évidence dans ces informations la présence d'une intrusion connue par la base de signature. L'efficacité de ce système de détection dépend donc fortement de la précision de sa base de signature. [35]

Il est possible d'élaborer des signatures plus génériques, qui permettent de détecter les variantes d'une même attaque, mais cela demande une bonne connaissance des attaques et du réseau, de façon à stopper les variantes d'une attaque et à ne pas gêner le trafic normal du réseau Une signature permet de définir les caractéristiques d'une attaque, au niveau des paquets (jusqu'à TCP ou UDP) ou au niveau protocole (HTTP, FTP…).

Au niveau paquet, l'IDS va analyser les différents paramètres de tous les paquets transitant et les comparer avec les signatures d'attaques connues. Au Niveau protocole, l'IDS va vérifier au niveau du protocole si les commandes Envoyées sont correctes ou ne contiennent pas d'attaque. Cette fonctionnalité a Surtout été développée pour HTTP actuellement. Néanmoins, plus il y a de signatures différentes à tester, plus le temps de traitement sera long, l'utilisation de signatures plus élaborées peut donc procurer un gain de temps appréciable. [33]

L'approche par scénario possède un certain nombre d'avantages et d'inconvénients :

Les avantages:

- Très efficace pour détecter des attaques sans produire un grand nombre de fausses alarmes.

- Peut rapidement et sûrement diagnostiquer l'utilisation d'un outil spécifique ou une technique d'attaque.
- Ceci peut aider les responsables de sécurité à donner la priorité aux mesures correctives.

Les inconvénients:

- Peut seulement détecter les attaques connues, dont les signatures sont introduites dans le système, donc le système de détection doit être constamment mis à jour avec les signatures des nouvelles attaques.
- Beaucoup de systèmes adoptant cette approche sont conçus pour employer un nombre limité de signatures qui peuvent être définis, ce qui les empêchent de détecter des variantes de ces attaques [34]

4.3 Types de réponses :

Une autre façon de classer les systèmes de détection d'intrusions consiste à les classer par type de réaction lorsqu'une attaque est détectée :

4.3.1 Réponse passive :

Un IDS passif est un système configuré uniquement pour surveiller et analyser l'activité du trafic réseau et pour alerter un opérateur des vulnérabilités et attaques éventuelles. Il n'est pas capable d'exécuter seul des fonctions de protection ou de correction.

Lorsqu'une attaque est détectée, ils génèrent une alarme et notifient l'administrateur système par e-mail, message dans une console, voir même par beeper. C'est alors lui qui devra prendre les mesures qui s'imposent. Les principaux avantages des systèmes IDS passifs sont que ces systèmes peuvent être déployés facilement et rapidement.

4.3.2 Réponse Active :

La réponse active au contraire a pour but de stopper une attaque au moment de sa détection. Pour cela on dispose de deux techniques : la reconfiguration du firewall et l'interruption d'une connexion TCP.

La reconfiguration du firewall permet de bloquer le trafic malveillant au niveau du firewall, en fermant le port utilisé ou en interdisant l'adresse de l'attaquant. Cette fonctionnalité dépend du modèle de firewall utilisé, tous les modèles ne permettant pas la reconfiguration

par un IDS. De plus, cette reconfiguration ne peut se faire qu'en fonction des capacités du firewall.

L'IDS peut également interrompre une session établie entre un attaquant et sa machine cible, de façon à empêcher le transfert de données ou la modification du système attaqué. Pour cela l'IDS envoie un paquet TCP reset aux deux extrémités de la connexion (cible et attaquant). Un paquet TCP reset a le flag RST positionné, ce qui indique une déconnexion de la part de l'autre extrémité de la connexion de chaque extrémité en étant destinataire, la cible et l'attaquant pensent que l'autre extrémité s'est déconnectée et l'attaque est interrompue.

Dans le cas d'une réponse active, il faut être sûr que le trafic détecté comme malveillant l'est réellement, sous peine de déconnecter des utilisateurs normaux. En général, les IDS ne réagissent pas activement à toutes les alertes. Ils ne répondent à des alertes que quand celles-ci sont positivement certifiées comme étant des attaques. L'analyse des fichiers d'alertes générés est donc une obligation pour analyser l'ensemble des attaques détectées.

Toutefois, il apparait que ce type de fonctionnalité automatique est potentiellement dangereux car il peut mener à des dénis de service provoqués par l'IDS. Un attaquant déterminé peut, par exemple, tromper l'IDS en usurpant des adresses du réseau local qui seront alors considérées comme la source de l'attaque par l'IDS. Il est préférable de proposer une réaction facultative à un opérateur humain (qui prend la décision finale).

Enfin, une dernière différenciation est la caractéristique d'utilisation qui Peut se faire d'une façon : continue (online) ou périodique (offline). [30]

4.4 Fréquence d'utilisation :

4.4.1 Utilisation continue :

La détection d'attaque se fait au moment où elle se produit, Dans la plupart des cas, avant d'attaquer un réseau, l'attaquant doit scanner l'environnement pour récolter des informations. Par conséquent, en voyant cela, on peut détecter une attaque avant même qu'elle ne se produise et ainsi y répondre le plus tôt possible. [31]

4.4.2 Utilisation périodique :

Cette méthode s'exécute périodiquement et on ne voit que le résultat d'attaque, elle est préférable pour avoir une défense plus fiable du point de vue du temps de calcul que pour la première utilisation. Contrairement à l'IDS online, l'attaque ne peut pas être détectée le plus

tôt possible pour l'éviter. Plus une attaque est détectée tardivement, plus les dommages sont importants. [31]

5. Conclusion :

Dans ce chapitre, nous avons présenté un état de l'art en matière de Détection d'intrusions. Pour bien positionner le problème nous avons d'abord Donné une définition à cette notion, ensuite nous avons présenté une classification des systèmes de détection d'intrusions (IDS) en se basant sur la source de données de ces derniers ainsi que sur leurs emplacements dans le système informatique. Cette classification comporte les IDS basés hôte (HIDS, les IDS basés sur l'application (AB-IDS) les IDS basés réseau (N-IDS). Nous avons abordé aussi les deux grandes approches utilisées par les Modes de détection IDS à savoir : La détection d'anomalies et La reconnaissance de signature Ensuite nous avons présenté les deux réponses apportées par les systèmes après détection d'une intrusion, que ce soit la réponse active qui englobe plusieurs techniques comme la reconfiguration du firewall et l'interruption d'une connexion TCP, ou bien la réponse passive dont le principe repose sur l'émission des alertes, la chose qui est assurée par la plupart des IDS actuels.

Pour conclure on peut dire que les IDS constituent une seconde ligne de Défense indispensable pour assurer la sécurité opérationnelle, surtout dans un Contexte d'interconnexion et d'ouverture croissant des systèmes informatiques.

Le chapitre suivant sera consacré à la deuxième phase de notre projet À savoir le machine Learning ainsi que des notions sur les données et les algorithmes d'apprentissage.

Chapitre 2 : Machine Learning

1. Introduction :

Ce domaine a pris toute son importance en raison de la révolution digitale des entreprises qui a conduit à la production de données massives de différentes formes et types, à des rythmes sans cesse en augmentation : le Big Data. [36]

Dans le domaine de l'intelligence artificielle, la machine Learning est la technologie qui permet aux machines d'apprendre seules à partir de données fournies. Ces algorithmes permettent aujourd'hui de résoudre des équations ou des cas qui semblaient être insolubles. [37]

Le machine Learning ou « apprentissage automatique » en français est un concept qui fait de plus en plus parler de lui dans le monde de l'informatique, et qui se rapporte au domaine de l'intelligence artificielle. Encore appelé « apprentissage statistique », ce terme renvoie à un processus de développement, d'analyse et d'implémentation conduisant à la mise en place de procédés systématiques. Pour faire simple, il s'agit d'une sorte de programme permettant à un ordinateur ou à une machine un apprentissage automatisé, de façon à pouvoir réaliser un certain nombre d'opérations très complexes.

L'objectif visé est de rendre la machine ou l'ordinateur capable d'apporter des solutions à des problèmes compliqués, par le traitement d'une quantité astronomique d'informations. Cela offre ainsi une possibilité d'analyser et de mettre en évidence les corrélations qui existent entre deux ou plusieurs situations données, et de prédire leurs différentes implications [38]

Le domaine d'application du machine Learning est très varié : la prédiction de valeurs financières, la détection d'intrusion dans le domaine de la sécurité informatique, le moteur de recherche influençable par le profil de l'utilisateur, la détection de vols de machine, l'implémentation d'un anti-virus et la cryptanalyse. Le cycle de vie d'une implémentation de la machine Learning est la suivante :

1. Obtention et nettoyage des données

2. Réalisation du modèle

3. Phase d'apprentissage

4. Phase de validation

5. Phase d'exécution

2 Cycle de vie d’une implémentation du machine Learning :

2.1 Obtention des données et pré-processing :

Comme le modèle est construit à partir des données, il est clair que plus on dispose de données, plus le modèle construit est précis et permettra ainsi de prendre de bonne décisions. Mais Quand les données d’un Data Set sont dans des ordres de grandeurs différents, certains algorithmes de Machine Learning mettent plus de temps à trouver un modèle prédictif optimal. [39]

Le prétraitement et le nettoyage des données sont alors des tâches importantes qui doivent intervenir avant d'utiliser un jeu de données à des fins d'apprentissage automatique. Les données brutes sont souvent bruyantes, peu fiables et incomplètes. Leur utilisation pour la modélisation peut générer des résultats trompeurs. [40]

La première étape à réaliser est donc l'obtention de données en suffisance, représentatives du problème à résoudre. En effet, s’il s’avère qu’on a beaucoup de données d’entraînement et/ou que l’algorithme d’apprentissage est lourd, il est possible qu’utiliser toutes les données prenne énormément de temps et/ou de ressources hardware. Dans ce cas, il faut naturellement échantillonner et ne récupérer qu’un petit pourcentage du dataset qui servira au travail de modélisation pour aller plus vite. Le problème lorsqu’on effectue un échantillonnage, c'est que l’on doit être bien sûr que cet échantillon est représentatif de toutes les données. On parle d'étape de *sampling* en anglais. [41]

La deuxième étape est le nettoyage, appelé aussi << pré-processing >> , de la donnée récoltée, cette étape consiste en un prétraitement de toutes ces données avant de commencer à les analyser : Gestion des valeurs manquantes, des valeurs textuelles (un algorithme sait gérer des chiffres et non pas des mots et il va s’agir de les “traduire” en chiffre), mise sous forme tabulaire des données ; le but de cette étape est de rendre la modélisation plus simple et rapide, d’optimiser le temps d’exécution et d’ apprentissage mais aussi d’extraire les caractéristiques des données pertinentes pour la prise de décision autrement appelées caractéristiques (features). Cette extraction de caractéristiques (features sélection) est une

bonne pratique, pour ne pas dire obligatoire, lors de la modélisation avec du Machine Learning. [42]
Il est conseillé de ne pas utiliser les mêmes données pour les phases de features sélections et d'évaluation, pour éviter un biais au niveau des performances estimées du système.

2.2 La modélisation et la mise en production :

Une fois les données prétraitées, il va s'agir de trouver le bon algorithme .De nombreux choix d'algorithmes d'apprentissage et de leurs hyper paramètres s'offrent aux Data Scientists. La nature du problème à résoudre permet en partie de guider ce choix. Ce choix doit donc être fait en fonction des résultats désirés ainsi que de la complétude des données [43]

Les facteurs qui peuvent entrer en jeu pour choisir le bon algorithme peuvent être nombreux, notamment le nombre de caractéristiques (*features),* la quantité de données qu'on a…etc. [44]

Une autre distinction qui nous aidera dans le choix d'un algorithme de machine Learning est le type de sortie que l'on attend de notre programme : est-ce une valeur **continue** (un nombre) ou bien une valeur **discrète** (une catégorie) ? Le premier cas est appelé une **régression**, le second une **classification**.

Néanmoins, il est nécessaire de savoir évaluer n'importe quel algorithme d'apprentissage sur son jeu de données Une évaluation rigoureuse des performances d'un algorithme est une étape indispensable à son déploiement. [43]

Voici un schéma permettant de vous aiguiller dans le choix de votre algorithme (voir figure 3)

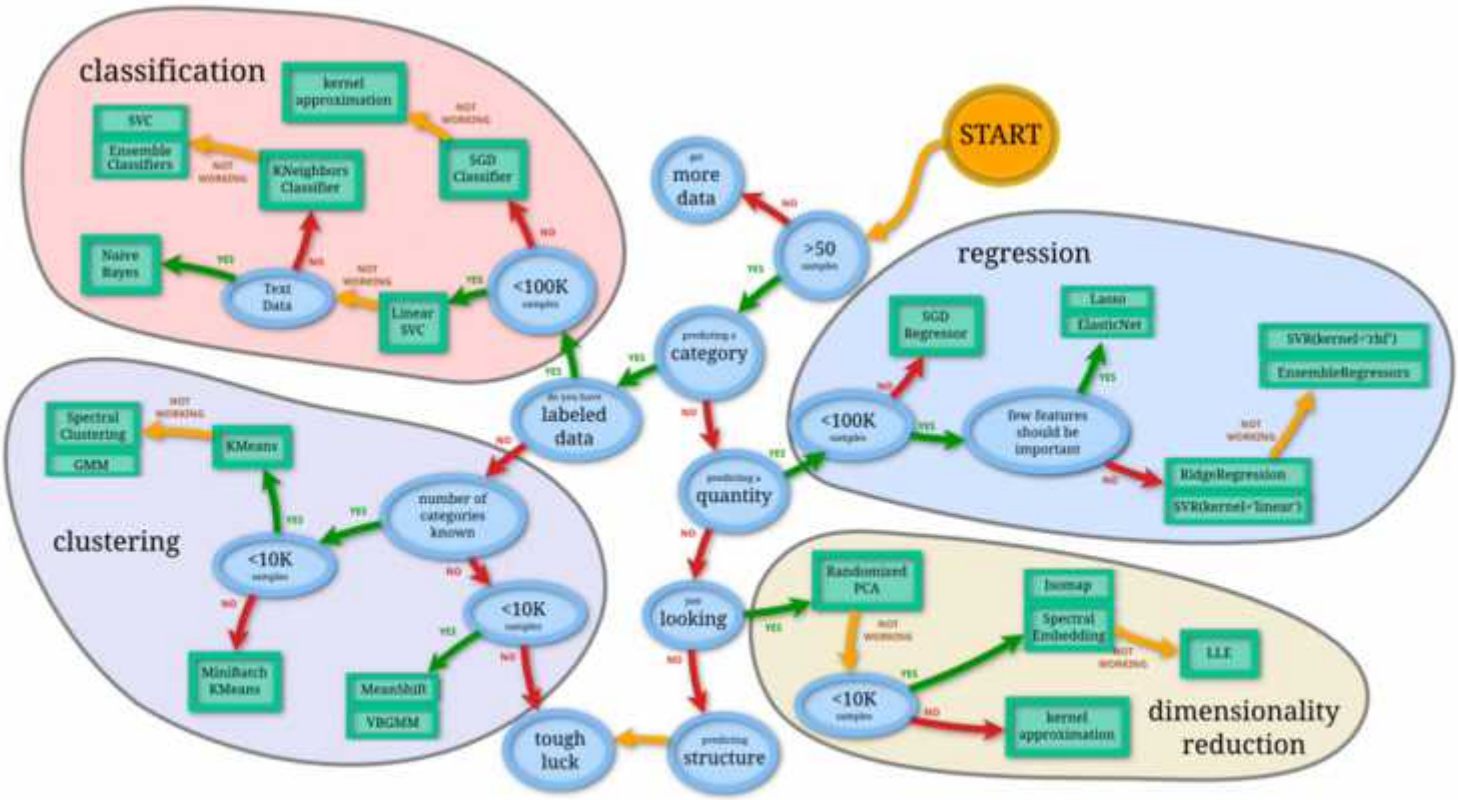


Figure 3 : Comment choisir l'algorithme à utiliser. [Online, 44]

2.3 Phase d'apprentissage :

En machine Learning, l'algorithme se construit une "représentation interne" afin de pouvoir effectuer la tâche qui lui est demandée (prédiction, identification, etc.). Pour cela, il va d'abord falloir lui entrer un jeu de données d'exemples afin qu'il puisse s'entraîner et s'améliorer, d'où le mot apprentissage**.** On sépare donc les données entre un jeu d'entraînement**,** sur lequel on apprend le modèle (Ce jeu de données s'appelle le training set.)Et un jeu de test**,** sur lequel on l'évalue**.** On peut appeler une entrée dans le jeu de données une instance ou une observation. [45]

2.3.1 Les différents procédés d'apprentissage :

Il existe principalement deux familles de méthodes d'apprentissage supervisé pour la classification : les méthodes discriminatives et les méthodes génératives.

Les méthodes discriminatives sont celles qui sont les plus utilisées et incluent, entre autres : SVM, KNN, régression logistique et réseaux de neurones. Elles permettent de définir une séparation entre les classes en créant des frontières de décision entre elles. La classification d'une nouvelle instance se fait selon la position de cette instance par rapport aux frontières de

décision. Les méthodes génératives visent, quant à elle, à apprendre des modèles qui décrivent la façon dont les données de chaque classe sont générées en considérant les classes indépendamment les unes des autres. Des exemples de cette famille de méthodes incluent les HMM, la classification naïve bayésienne, etc. [46]

La machine Learning implique deux principaux systèmes d'apprentissage qui définissent ses différents modes de fonctionnement. Il s'agit de :

a) L'apprentissage supervisé ou analyse discriminatoire

Ici, la machine s'appuie sur des classes prédéterminées et sur un certain nombre de paradigmes connus pour mettre en place un système de classement à partir de modèles déjà catalogués. Dans ce cas, deux étapes sont nécessaires pour compléter le processus, à commencer par le stade d'apprentissage qui consiste à la modélisation des données cataloguées. Ensuite, il s'agira au second stade de se baser sur les données ainsi définies pour attribuer des classes aux nouveaux modèles introduits dans le système, afin de les cataloguer eux aussi. [36]

b) L'apprentissage non-supervisé ou clustering

Dans ce mode de fonctionnement du machine Learning, il n'est pas question de s'appuyer sur des éléments prédéfinis, et la tâche revient à la machine de procéder toute seule à la catégorisation des données. Pour ce faire, le système va croiser les informations qui lui sont soumises, de manière à pouvoir rassembler dans une même classe les éléments présentant certaines similitudes. Ainsi, en fonction du but recherché, il reviendra à l'opérateur ou au chercheur de les analyser afin d'en déduire les différentes hypothèses. [36]

2.4 Phase de validation :

Durant cette phase, on va tester et valider le modèle et ses paramètres selon des critères se basant sur ses résultats. Il permet d'obtenir le meilleur modèle généralisant les données obtenues lors de la phase d'apprentissage. Pour cela, on a un ensemble d'exemples pour l'apprentissage et un autre pour les tests. Voici quelques méthodes pour les tests :

1. **Hold-out :** On coupe aléatoirement l'ensemble des informations en deux groupes : groupe d'apprentissage et groupe de tests.

2. **Leave-one-out :** Cette méthode sort de l'ensemble des informations une donnée en particulier et la laisse de côté, puis construit le modèle avec celles restantes et on évalue la

structure avec l'exemple laissé de côté. On répète le processus pour chacune des données de l'ensemble de données. Ainsi, on peut avoir une moyenne globale de la précision du modèle.

3. **Cross-validation :** Cette méthode réalise un partitionnement des données de manière aléatoire en n groupes. On utilise une partition comme un ensemble de test et le reste pour former celui d'entraînement. Comme précédemment, on applique un algorithme à l'ensemble d'entraînement et on évalue le modèle résultant sur celui de tests. On répète ce processus pour chaque partition et on regarde l'erreur moyenne. (Cette méthode sera celle que nous allons utiliser par la suite). [44]

Après avoir validé le modèle, il reste à quantifier ses performances en pratique.

2.5 Performance du modèle

Après avoir déroulé son algorithme sur ses données d'entraînement *(Training set)* et faire des prédictions avec le jeu de test *(Test Set)*, il est temps d'évaluer la performance de notre algorithme.

On mesure la performance du modèle sur la base de test. Il existe certaine façon standard de le faire en fonction des types de problèmes : [47]

Classification :

- matrice de confusion
- courbe ROC
- précision / rappel

Régression :

- erreur de prédiction
- graphe XY valeur à prédire / valeur prédite

Clustering :

- variance intra classe, inter classe
- nombre d'arc coupés

Les métriques de détection d'intrusion aident à évaluer les performances d'un système de détection d'intrusion. Les mesures d'évaluation les plus couramment utilisées pour la détection d'intrusion sont les suivantes: False Alarm Rate (FAR), Détection Rate (DR), Rappel, précision, spécificité, F-score.

Toutes ces métriques d'évaluation sont essentiellement dérivées des quatre attributs de base de la matrice de confusion décrivant les classes réelles et prévues. Ces éléments de la matrice de confusion sont:

True Negative (TN): nombre d'instances correctement prédites comme normal.

False Négative (FN): nombre d'instances prédites à tort comme normal

Faux Positif (FP) : nombre d'instances prédites à tort comme des attaques.

True Positive (TP) : nombre d'instances correctement prédites comme des attaques.

2.6 Types de modèles :

Il existe trois manières de faire du machine Learning :

a) Par régression:

Ce type d'algorithmes supervisés va trouver une valeur continue (un nombre réel) qui est la prédiction de la valeur d'une nouvelle observation donnée. Par exemple la prédiction de prix de maison en fonction de ses caractéristiques.

b) Par classification:

Ces algorithmes vont classifier une donnée dans une catégorie. Par exemple classifier un courriel en SPAM ou non, décider si une tumeur est maligne ou bénigne. Il s'agit également d'un algorithme supervisé.

c) Par clustering:

Il s'agit d'une famille d'algorithmes non supervisés. Par conséquent, les données n'ont pas d'étiquettes (Non-labeled Data). Il va regrouper les données par similarité. Par exemple, on fournit un ensemble de photos d'animaux (sans qu'on dise de quels animaux il s'agit). Enfin il va regrouper les photos de chats d'un coté et celle des chiens d'un autre etc.

Puisque notre problème semble être un problème de classification supervisé, ce qui suit est une liste non exhaustive de modèles de classification qui pourraient être envisagées pour la mise en place de notre IDS.

3. Algorithme de classification :

La classification est une méthode d'exploration de données permettant d'affecter des instances de données à l'une des rares catégories. Il existe de nombreux algorithmes de classification développés pour se surpasser. Ils fonctionnent tous selon des techniques mathématiques telles que l'arbre de décision, la programmation linéaire et les réseaux de neurones. Ces techniques analysent les données disponibles de plusieurs manières pour en faire la prévision. Voici ceux que nous allons utilisés dans notre projet :

3.1 Naïve bayes :

C'est une technique de classification basée sur le théorème de Bayes avec une hypothèse d'indépendance parmi les prédicateurs. En termes simples, un classificateur Naïve Bayes suppose que la présence d'une fonctionnalité particulière dans une classe n'est pas liée à la présence d'une autre fonctionnalité. Par exemple, un fruit peut être considéré comme une pomme s'il est rouge, rond et d'environ 3 pouces de diamètre. Même si ces caractéristiques dépendent les unes des autres ou de l'existence d'autres caractéristiques, toutes ces propriétés contribuent indépendamment à la probabilité que ce fruit soit une pomme et c'est pourquoi il est appelé «naïf».

Le modèle Naïve Bayes est facile à construire et particulièrement utile pour les très grands ensembles de données. Outre sa simplicité, Un de ces points fort est le besoin d'une faible quantité d'informations pour la phase d'apprentissage. Naïve Bayes est réputé a surpasser même les méthodes de classification les plus sophistiquées

Bayes theorem:

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

Équation 1 : Theorem de Naive Bayes.

Le terme P (A \ B) se lit **:** la probabilité que l'événement A se réalise sachant que l'événement B s'est déjà réalisé. https://mrmint.fr/naive-bayes-classifier

$$P(A) = \frac{c_i \qquad (A)}{c_i \qquad t_i}$$

$$P(B \cap A) = \frac{c_{(B \cap A)}}{c_{tot}}$$

Note:

- le cardinal d'un ensemble est le nombre d'éléments dans ce dernier
- cardinal total représente l'ensemble total des instances

Cas ou on a plusieurs paramètres / caractéristiques :

$$P(y|x_1, \ldots, x_n) = \frac{P(x_1|y)P(x_2|y)\ldots P(x_n|y)P(y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

Équation 2 Naive bayes a plusieurs carateristiques.

Avantage :

- le Naïve Bayes Classifier est très rapide pour la classification : en effet les calculs de probabilités ne sont pas très coûteux.
- La classification est possible même avec un petit jeu de données

Inconvénients :

-

Contre intuitivement, malgré la violation de la contrainte d'indépendance des variables, Naïve Bayes donne de bons résultats de classification. [49]

3.2 Arbre De décision :

Cette technique divise le problème de classification en sous-problèmes. Il crée un arbre de décision qui est utilisé pour développer un modèle utilisé aux fins de la classification. Effectuer une classification est relativement simple. Il suffit de répondre aux questions en descendant dans l'arbre de décision

Les arbres de décision sont des modèles **non-paramétriques** et trouvent des règles en général assez puissantes. Ils peuvent traiter des très grands dataset et ils peuvent aussi utiliser des prédicateurs mixtes (catégoriques et nombres). Les variables redondantes sont éliminées, parce que l'arbre ne les prend même pas en compte.

A chaque nœud, on doit donc trouver deux choses : quelle features utiliser et quel est le point de séparation des deux zones, de sorte à minimiser l'erreur quadratique pour la régression et l'impureté pour la classification. On fait ainsi grandir l'arbre de la manière la plus grande possible

Les forêts aléatoires sont donc un ensemble d'arbres de décisions entraînés individuellement, légèrement différents les uns des autres. Pour prédire une nouvelle valeur, on effectue la classification pour chaque arbre de cette forêt. La forêt choisit la valeur ayant le plus de votes parmi tous ses arbres.

Pour résumer, les forêts aléatoires sont nommées ainsi parce qu'**on injecte de l'aléatoire dans la création de chaque arbre**. Premièrement avec la création d'échantillons aléatoires (bootstraping). Deuxièmement, avec la sélection des features sur lesquels effectuer le test de séparation.

Avantage :

Les forêts aléatoires sont un algorithme possédant de très bonnes performances assez facilement sur la plupart des problèmes que vous rencontrerez. C'est un *go-to* à tester dès le départ en abordant un problème de modélisation, sans trop besoin de faire de features engineering en amont. De plus, il fonctionne bien sur les grosses bases de données, car il ne possède pas une si grosse complexité pour l'entraînement. Ceci-dit si vous êtes en présence d'un dataset volumineux, vous pouvez effectuer une parallélisations des calculs sur plusieurs CPU (avec l'argument n_jobs dans scikit-learn).

Encore une fois, ce modèle est rapide. En plus, les forêts n'overfittent pas. Le problème de taille mémoire pour de gros jeux de données est le stockage du jeu de données lui-même [50]

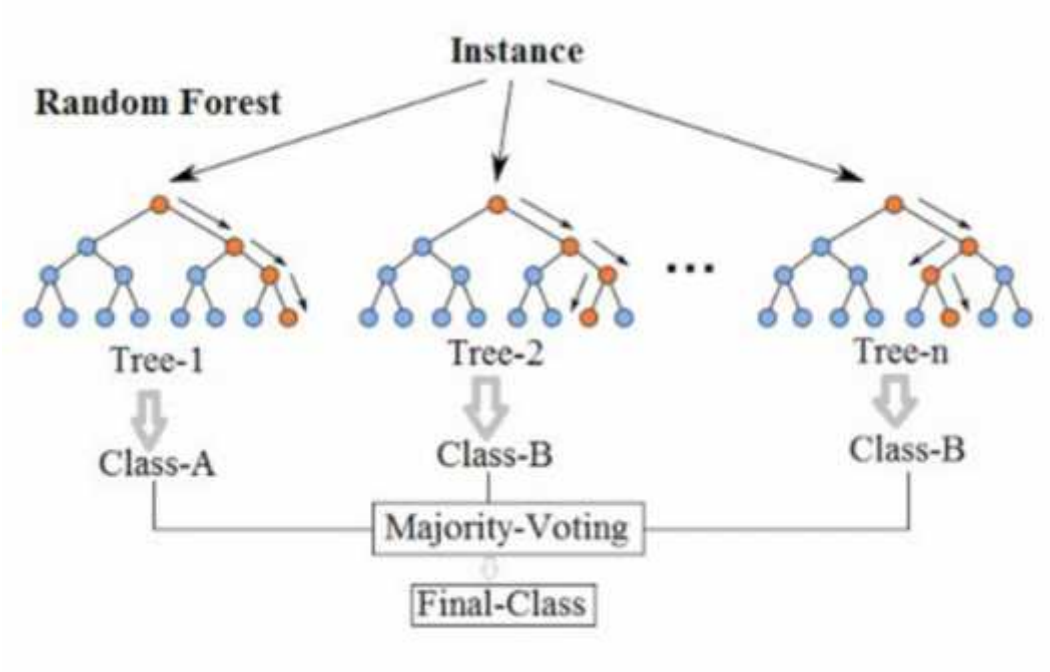


Figure 4 : algorithme des arbres de decision.[Online,50]

3.3 Machine à vecteur de support (SVM) :

Le SVM appartient à la catégorie des classificateurs linéaires (qui utilisent une séparation linéaire des données), et qui dispose de sa méthode à lui pour trouver la frontière entre les catégories.

Pour que le SVM puisse trouver cette frontière, il est nécessaire de lui donner des données d'entraînement. En l'occurrence, on donne au SVM un ensemble de points, dont on sait déjà si ce sont des carrés rouges ou des ronds bleus, comme dans la Figure 5. A partir de ces données, le SVM va estimer l'emplacement le plus plausible de la frontière : c'est la période d'entraînement, nécessaire à tout algorithme d'apprentissage automatique.

Une fois la phase d'entraînement terminée, le SVM a ainsi trouvé, à partir de données d'entraînement, l'emplacement supposé de la frontière. En quelque sorte, il a « appris » l'emplacement de la frontière grâce aux données d'entraînement. Qui plus est, le SVM est maintenant capable de prédire à quelle catégorie appartient une entrée qu'il n'avait jamais vue avant, et sans intervention humaine (comme c'est le cas avec le triangle noir dans la Figure 5): c'est là tout l'intérêt de l'apprentissage automatique. [51]

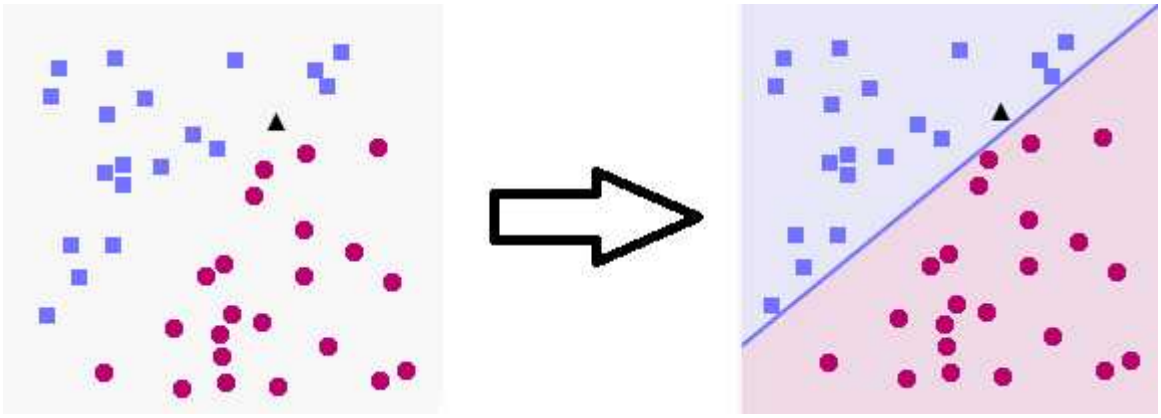


Figure 5 : utilisation de l'algorithme SVM. [51]

Quand on a un ensemble de points d'entraînement, il existe plusieurs lignes droites qui peuvent séparer nos catégories. La plupart du temps, il y en a une infinité… Alors, laquelle choisir ?

C'est simple, un SVM va placer la frontière aussi loin que possible des carrés bleus, mais également aussi loin que possible des ronds rouges.

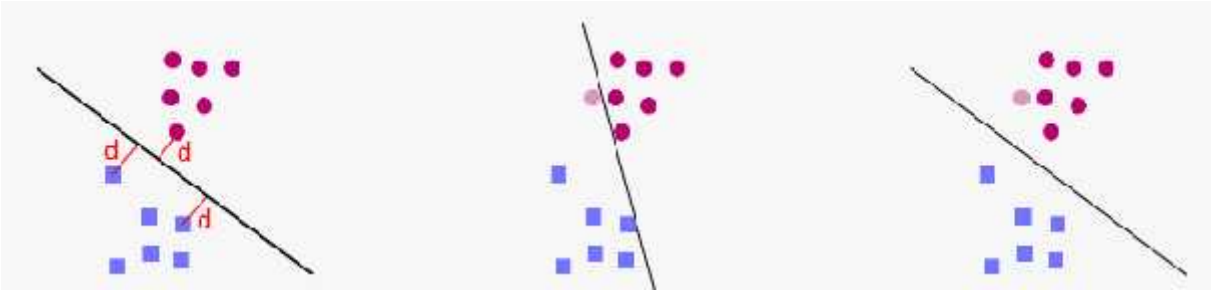


Figure 6 : emplacement de la frontière. [51]

Comme on le voit dans la figure 6, A gauche, on maximise la distance dd entre la frontière et les points d'entraînement. Au centre, une frontière non-optimale, qui passe très près des points d'entraînement, cette frontière classe de façon erronée le rond rouge clair comme un carré bleu ! Au contraire, à droite, la frontière optimale (qui maximise dd) classe bien le rond rouge clair comme un rond rouge

On conclue que c'est bien la frontière la plus éloignée de tous les points d'entraînement qui est optimale, on dit qu'elle a la meilleure capacité de généralisation. Ainsi, le but d'un SVM est de trouver cette frontière optimale, en maximisant la distance entre les points d'entraînement et la frontière.

Cependant, il est considéré que, étant donné un ensemble de données d'entraînement, les SVM sont des outils qui obtiennent parmi les meilleurs résultats. En fait, il a même été prouvé

que dans la catégorie des classificateurs linéaires, les SVM sont ceux qui obtiennent les meilleurs résultats.

Un des autres avantages des SVM, et qu'il est important de noter, est que ces derniers sont très efficaces quand on ne dispose que de peu de données d'entraînement : alors que d'autres algorithmes n'arriveraient pas à généraliser correctement, on observe que les SVM sont beaucoup plus efficaces. Cependant, quand les données sont trop nombreuses, le SVM a tendance à baisser en performance. [51]

3.4 L'algorithme des K plus proche voisin (KNN) :

L'algorithme des k-plus proches voisins (KNN) est une méthode d'apprentissage à base d'instances. Il ne comporte pas de phase d'apprentissage en tant que telle. Cette méthode enregistre toutes les classes qui lui sont fournies à l'aide de données d'apprentissage, définit et classe les nouvelles instances en fonction d'une mesure de similarité. Ses k plus proches voisins sont alors considérés : on observe leur classe/catégorie, et celle qui revient le plus (La classe majoritaire) parmi les voisins est affectée aux nouvelles instances à classer.

La distance peut, en général, être n'importe quelle mesure métrique: la distance euclidienne standard est le choix le plus courant. Les méthodes basées sur les voisins sont connues sous le nom de méthodes d'apprentissage automatique non généralisantes, car elles «se souviennent» simplement de toutes ses données d'apprentissage

Données en entrée :

- un ensemble de données D.
- une fonction de définition distance d
- Un nombre entier K

Pour une nouvelle observation X dont on veut prédire sa variable de sortie y Faire :

1. Calculer toutes les distances de cette observation X avec les autres observations du jeu de données D
2. Retenir les K observations du jeu de données D les proches de X en utilisation la fonction de calcul de distance d
3. Prendre les valeurs de y des K observations retenues :

1. Si on effectue une régression, calculer la moyenne (ou la médiane) de y retenues
2. Si on effectue une classification, calculer le mode de y retenues

4. Retourner la valeur calculée dans l'étape 3 comme étant la valeur qui a été prédite par K-NN pour l'observation X. [52]

3.4.1 Calcul de similarité dans l'algorithme K-NN

Il existe plusieurs fonctions de calcul de distance, notamment, la distance euclidienne, la distance de Manhattan, la distance de Minkowski …etc. On choisit la fonction de distance en fonction des types de données qu'on manipule. Ainsi pour les données quantitatives (exemple: poids, salaires, taille, montant de panier électronique etc.…) et du même type, la distance euclidienne est un bon candidat. Quant à la distance de Manhattan, elle est une bonne mesure à utiliser quand les données (input variables) ne sont pas du même type (exemple : âge, sexe, longueur, poids etc.…).

Il est inutile de coder, soi-même ces distances, généralement, les librairies de Machine Learning comme Scikit Learn, effectue ces calculs en interne. Il suffit juste d'indiquer la mesure de distance qu'on souhaite utiliser. [52]

3.4.2 Calcul de distance :

a) La distance euclidienne:

- distance qui calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points :

$$D_e(x,y) = \sqrt{\sum_{j=1}^{n}(x_j - y_j)^2}$$

Équation 3: la distance euclidienne.

b) La distance de Manhattan :

- la distance de Manhattan: calcule la somme des valeurs absolues des différences entre les coordonnées de deux points :

$$D_m(x,y) = \sum_{i=1}^{k} | x_i - y_i |$$

Équation 4: Distance de Manhattan.

3.4.3 Choix de K :

Le choix de la valeur k à utiliser pour effectuer une prédiction avec K-NN, varie en fonction du jeu de données. En règle générale, moins on utilisera de voisins (un nombre K petit) plus on sera sujette au sous apprentissage (underfitting). Par ailleurs, plus on utilise de voisins (un nombre K grand) plus, sera fiable dans notre prédiction. Toutefois, si on utilise nombre de voisins avec K = N et N étant le nombre d'observations, on risque d'avoir du overfitting et par conséquent un modèle qui se généralise mal sur des observations qu'il n'a pas encore vu.

Exemple : dans l'exemple suivant, si K=3 il (le ?) sera classé avec les croix, si K=5 il sera classé avec les rond [52]

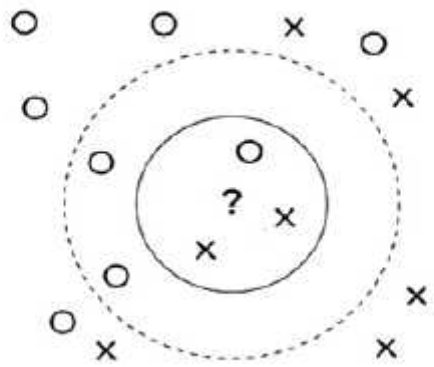


Figure 7 : Algorithme KNN.

Toutes les méthodes discutées sont connues pour leurs traits saillants et leurs inconvénients inhérents. Il faut du temps pour que l'arbre de décision soit construit. La méthode du voisin le plus proche prend énormément de temps lorsque la taille de l'ensemble de données augmente. Ne fonctionne mieux que sur des données numériques, ce qui nécessite la conversion des données textuelles de l'ensemble de données en une valeur numérique.

4. CONCLUSION

La mise en œuvre de solutions de machine Learning est grandement facilitée par les implémentations open source des différents algorithmes. Il n'est alors plus nécessaire de connaître le fonctionnement exact de ces derniers ainsi que les mathématiques poussées dont ils proviennent.
Néanmoins, une bonne connaissance des données sur lesquelles on travaille est nécessaire pour concevoir des modèles efficaces.

Il est facilement oublié que le machine Learning est l'étape finale du processus d'analyse des données. Avant de considérer faire des prédictions, il faut déjà savoir collecter les données, les prétraiter (car les sources sont bien souvent inconstantes), les stocker, les visualiser pour mieux les connaître et enfin seulement il est intéressant d'envisager le machine learning. Le prochain chapitre va expliquer l'utilisation du machine Learning dans le domaine des détections d'intrusions : le dataset utilisé ainsi que la méthodologie de notre travail.

Chapitre 3 : Conception et mise en Œuvre

1. Introduction :

Les IDS basés sur les techniques de classifications ont pour but de classifier les trafics d'un réseau en deux classes : « général » et «intrusion », la classification nécessite un apprentissage. La précision de cet apprentissage assure la diminution du taux de faux positifs (i.e. les cas normaux classés comme intrusions) et du taux de faux négatifs (i.e. les intrusions classées comme normales/non-attaque)

Ce chapitre présente la description de notre solution proposée dans le cadre de ce PFE, cette dernière comprend deux majeurs étapes : la première consiste en l'implémentation de notre Ids afin que les informations du paquet réseau sniffé soit compatibles avec celles se trouvant dans notre dataset, qui par ailleurs est le NSL KDD.

La deuxième étape quant a elle, se résume en l'utilisation des algorithmes de classification afin que ces derniers nous aident à définir/prédire si le paquet réseau sniffé est un paquet normal/non-attaque ou bien qu'il s'agit d'une attaque.

2. Conception

Le schéma global du fonctionnement de notre IDS est décrit par les diagrammes de cas d'utilisation suivants:

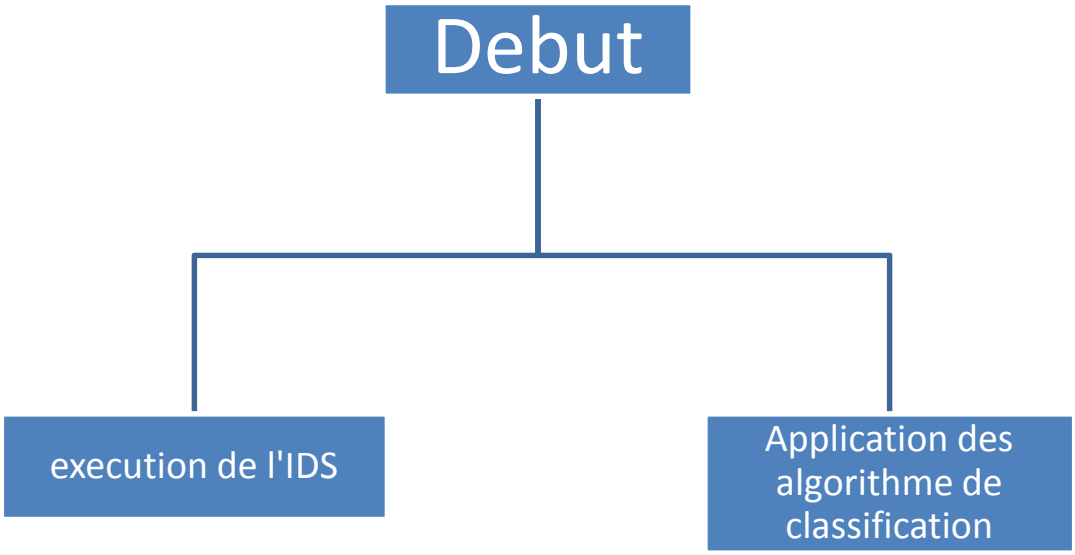


Figure 8 : notre solution.

Le fonctionnement globale de notre IDS consiste premierement en l'execution de notre IDS(que nous allons voir et expliquer dans la figure 9) ensuite viens l'etape du machine learning : l'application des algorithmes de classification (que nous detailons dans la figure 10)

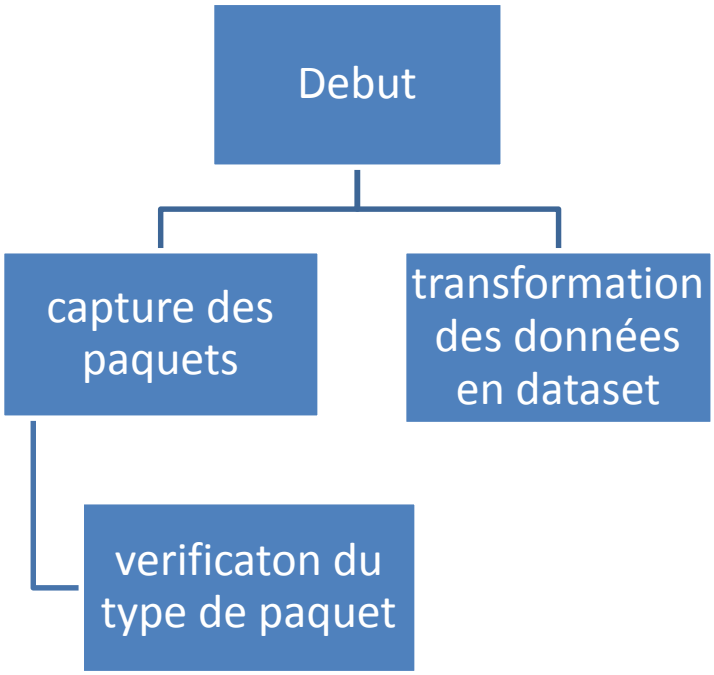


Figure 9 : Module IDS

Comme nous le montre la figure 9, l'étape d'exécution de l'ids consiste en premier à capturer/sniffer les paquets réseaux (nous avons intégré plusieurs type de paquets à capturer : TCP , UDP ou même IP) et à transformer ensuite les données reçues par l'ids en dataset (test) , afin que ce dernier soit compatible avec le dataset d'entrainement (Train).

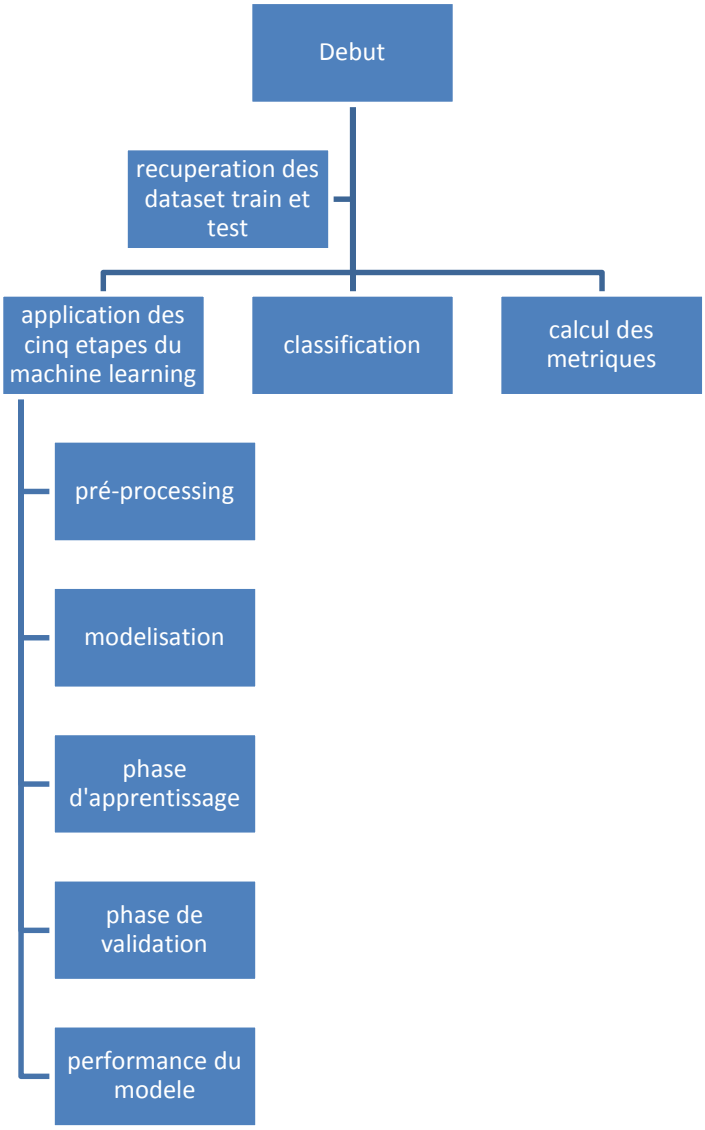


Figure 10 : classification.

Viens ensuite l'étape de l'utilisation des algorithmes de classification, pour cela, on doit tout d'abord passer par les cinq étapes du machine Learning, étant donné que les attributs des dataset récupérés ne sont pas tous numériques, il se peut aussi qu'il y est des données redondantes, c'est pour cela que nous devrons passer par ces étapes de nettoyage et pré processing afin que les données du dataset soit utilisables par les algorithme de classification. Une fois que les deux dataset sont nettoyé et prêt a être utiliser, viens alors l'étape de classification , pour prédire si le paquet capturé s'agit d'une attaque une d'un paquet classé comme normal (à l'aide des algorithme de classification). Une fois terminé, nous avons procédé à un calcul de métriques afin d'évaluer lequel des algorithmes utilisé est le plus performant.

3. Outils de réalisation :

Dans cette partie nous allons présenter les principaux outils utilisés pour la mise en place de notre projet.

3.1 Choix de langage de programmation python :

Les modules conçus ont été réalisés sous python 2.7 (software fondation, crée en 2001), ces principales caractéristiques sont résumées dans les points suivant :

- Le langage python est Portable : un Système d'intrusion ne doit ni dépendre de l'architecture matérielle ni du système d'exploitation.
- est un langage de programmation interprété
- est un des langages principaux utilisés en data analysis (analyse de donnée) et en machine Learning (apprentissage par la machine)
- Peut créer des interfaces graphiques.
- Peut Faire circuler des informations à travers un réseau.

Nous avons utilisé scikit-learn qui est une bibliothèque d'apprentissage statique en python, c'est le moteur de beaucoup d'application de l'intelligence artificielle et de la science des données, elle est construite a base de :

- Numpy (version utilisé 1.16.3) : extension de python, elle permet de créer directement un tableau depuis un fichier ou au contraire de sauvegarder un tableau dans un fichier, et manipuler des vecteurs, matrices et polynômes.
- Scipy : Cette distribution de modules est destinée à être utilisée afin de créer un environnement de travail scientifique très similaire à celui offert par MATLAB, ou R. Il contient des modules pour l'optimisation, l'algèbre linéaire, les statistiques et les traitements d'images.
- Matplotlib (version utilisée 2.1.2) : est une bibliothèque de traçage de graphique python 2D capable de générer des histogrammes, des spectres de puissance, des graphiques a barre, différents diagrammes ave seulement quelques ligne de code

4. Méthodologie de recherche

Les étapes suivies / outils utilisés dans le cadre de la méthodologie de recherche sont les suivantes:

- Un ordinateur Asus Intel® Pentium® CPU N 3700 ayant 4Go de Ram est utilisé pour la démonstration de ce projet.(cela explique la longue durée d'exécution = 3H30)
- Python 2.7 est utilisé comme langage de programmation.
- Le jeu de données NSL KDD est sélectionné.
- Scikit learn est choisi pour la simulation (Pandas : 0.24.2, Numpy : 1.16.3, Matplotlib : 2.1.2, Sklearn : 0.20.3, imblearn : 0.4.3)
- Knn, naïve bayes, svm, Radom Forest, décision tree sont utilisés comme classifier dans notre projet et classent les instances en attaque ou en normale.
- Le prétraitement du fichier de training et de test avec 42 attributs est effectué afin d'être utilisé pour les calculs associés a chaque algorithme de classification.

5. Choix du Dataset :

Les systèmes intelligents de détection d'intrusion ne peuvent être construits que s'il existe un ensemble de données efficace. Un ensemble de données contenant une quantité appréciable de données de qualité imitant le temps réel ne peut que permettre de former et de tester un système de détection d'intrusion. L'ensemble de données NSL-KDD est une version perfectionnée de l'ensemble de données KDD 99 précédent. Dans ce projet, le jeu de données NSL-KDD est analysé et utilisé pour étudier l'efficacité des différents algorithmes de classification dans la détection des anomalies dans les modèles de trafic réseau.

L'ensemble de données NSL-KDD avec 42 attributs est une amélioration par rapport à KDD'99 data set duquel les instances en double ont été supprimées pour éliminer les résultats de classification biaisés.

Différentes configurations de cet ensemble de données sont disponibles avec des variations de nombre d'instances, mais le nombre d'attributs dans chaque cas est 42. L'attribut appelé 42 dans l'ensemble de données est l'Attribut «class» indiquant si une instance donnée est une instance de connexion normale ou une attaque. Le tableau 1 donne la description des attributs de fichier KDD avec l'explication de chaque attribut.

Nr	Features	
	Name	Description
1	duration	duration of connection in seconds
2	protocol_type	connection protocol (tcp, udp, icmp)
3	service	dst port mapped to service (e.g. http, ftp, ..)
4	flag	normal or error status flag of connection
5	src_bytes	number of data bytes from src to dst
6	dst_bytes	bytes from dst to src
7	land	1 if connection is from/to the same host/port; else 0
8	wrong_fragment	number of 'wrong' fragments (values 0,1,3)
9	urgent	number of urgent packets
10	hot	number of 'hot' indicators (bro-ids feature)
11	num_failed_logins	number of failed login attempts
12	logged_in	1 if successfully logged in; else 0
13	num_compromised	number of 'compromised' conditions
14	root_shell	1 if root shell is obtained; else 0
15	su_attempted	1 if 'su root' command attempted; else 0
16	num_root	number of 'root' accesses
17	num_file_creations	number of file creation operations
18	num_shells	number of shell prompts
19	num_access_files	number of operations on access control files
20	num_outbound_cmds	number of outbound commands in an ftp session
21	is_hot_login	1 if login belongs to 'hot' list (e.g. root, adm); else 0
22	is_guest_login	1 if login is 'guest' login (e.g. guest, anonymous); else 0
23	count	number of connections to same host as current connection in past two seconds
24	srv_count	number of connections to same service as current connection in past two seconds
25	serror_rate	% of connections that have 'SYN' errors
26	srv_serror_rate	% of connections that have 'SYN' errors
27	rerror_rate	% of connections that have 'REJ' errors
28	srv_rerror_rate	% of connections that have 'REJ' errors
29	same_srv_rate	% of connections to the same service
30	diff_srv_rate	% of connections to different services
31	srv_diff_host_rate	% of connections to different hosts
32	dst_host_count	count of connections having same dst host
33	dst_host_srv_count	count of connections having same dst host and using same service
34	dst_host_same_srv_rate	% of connections having same dst port and using same service
35	dst_host_diff_srv_rate	% of different services on current host
36	dst_host_same_src_port_rate	% of connections to current host having same src port
37	dst_host_srv_diff_host_rate	% of connections to same service coming from diff. hosts
38	dst_host_serror_rate	% of connections to current host that have an S0 error
39	dst_host_srv_serror_rate	% of connections to current host and specified service that have an S0 error
40	dst_host_rerror_rate	% of connections to current host that have an RST error
41	dst_host_srv_rerror_rate	% of connections to the current host and specified service that have an RST error
42	connection_type	

Tableau 1 : attributs du dataset NSL KDD.

L'analyse du jeu de données NSL-KDD est réalisée à l'aide de divers algorithmes de classification disponibles dans l'outil d'exploration de données **scikit-learn.** L'ensemble de données NSL-KDD est analysé et classé en quatre groupes différents décrivant les 4 types d'attaques les plus courants (DOS, U2R, R2L, probe). Une étude analytique approfondie est réalisée sur le jeu de données de test et de training.

Les classes d'attaques présentes dans l'ensemble de données NSL-KDD sont regroupées en quatre catégories :

5.1 Catégorie d’attaques dans le Dataset :

5.1.1 DOS:

Le déni de service est une catégorie d'attaque, qui épuise les ressources de la victime, l'empêchant ainsi de traiter des demandes légitimes. -par exemple syn. inondation. Caractéristiques pertinantes: “source bytes” and “percentage of packets with errors”»

5.1.2 Objet de probation:

L’objectif de la surveillance et vérification des autres attaques est d'obtenir des informations sur la victime distante, par exemple. Balayage de port.
Fonctionnalités pertinentes: “duration of connection” and “source bytes”

5.1.3 U2R:

L’accès non autorisé aux privilèges super utilisateur (root) locaux est un type d'attaque, par lequel un attaquant utilise un compte normal pour se connecter au système victime et tente de gagner du temps. Les privilèges root / administrateur en exploitant une vulnérabilité de la victime, par exemple tampons par débordement attaqués. Caractéristiques pertinantes: “number of file creations” and “number of shell prompts invoked.

5.1.4 R2L:

Accès non autorisé depuis une machine distante, l'attaquant s'introduit dans une machine distante et obtient un accès local à la machine victime. Par exemple : Deviner le mot de passe. Caractéristiques pertinentes: Network level features –“duration of connexion” and “service requested” and host level features -“number of failed login attempts”

Chaque types d’attaques et subdivisée en sous attaques présenté dans le tableau 2.

Normal	Probing	DOS	R2L	U2R
Normal (97277)	Nmap (231)	Land (21)	Spy(2)	Buffer_over flow(30)
	Portswee p (1040)	Pod (264)	Phf(4)	Rootkit(10)
	Ipsweep (1247)	Teardrop (979)	Multihop (7)	Loadmodul e (9)
	Satan (1589)	Back (2203)	ftp_write (8)	Perl(3)
		Neptune (107201)	Imap(12)	
		Smurf (280790)	Warezmas ter(20)	
			Guess_ passwd (53)	

Tableau 2 : Différents types d'attaques du dataset.

Attack Class / Protocol	DoS	Probe	R2L	U2R
TCP	42188	5857	995	49
UDP	892	1664	0	3
ICMP	2847	4135	0	0

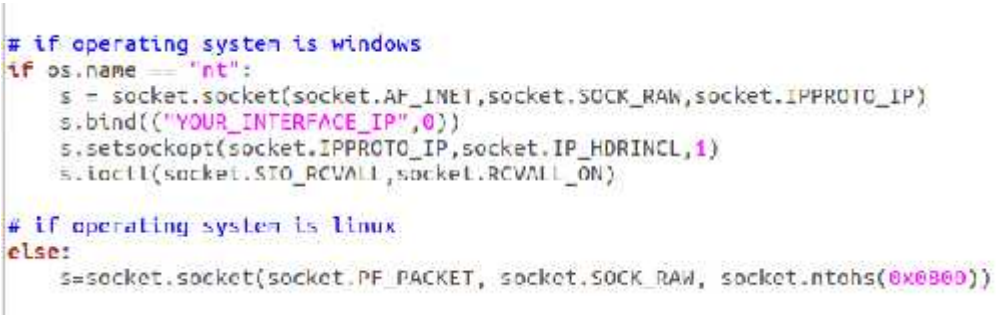
Tableau 3 : Protocole Utilisé pour les différents types d'attaques.

6. Implementation de l'ids:

Pour la création de notre Ids, nous avons opté pour l'implémentation d'un module en python capable d'extraire les informations nécessaires et essentielles à la création de notre base de test afin que cette dernière soit compatible avec notre base de connaissance (training dataset).

Nous avons utilisé pour cela un module de socket (qu'il faudra importer au début) : en langage de programmation python, le module socket nous permet de jouer avec le concept de réseau. Donc ici pour capturer les paquets, nous allons utiliser le module socket. Socket.

Pour renifler avec un module de socket en python, nous devons créer un objet de classe socket. Socket avec une configuration spéciale. En termes simples, nous devons configurer l'objet de classe socket. Socket pour capturer les paquets de bas niveau du réseau, de manière à ce qu'il puisse capturer les paquets de réseaux de bas niveau et nous fournisse une sortie sans faire aucun type de modification des paquets de capture.
Il existe une petite différence entre les codes de module de socket Python basés sur les systèmes d'exploitation. Parce que le noyau Windows fonctionne différemment du noyau Linux.



```
# if operating system is windows
if os.name == "nt":
    s = socket.socket(socket.AF_INET,socket.SOCK_RAW,socket.IPPROTO_IP)
    s.bind(("YOUR_INTERFACE_IP",0))
    s.setsockopt(socket.IPPROTO_IP,socket.IP_HDRINCL,1)
    s.ioctl(socket.SIO_RCVALL,socket.RCVALL_ON)

# if operating system is linux
else:
    s=socket.socket(socket.PF_PACKET, socket.SOCK_RAW, socket.ntohs(0x0800))
```

Figure 11 : Ecoute sur le réseau.

6.1 Comment analyser / extraire les paquets capturés?

Il existe différents types de formats de données disponibles dans la mise en réseau nous allons donc seulement décrire quelques formats de données importants et les plus utilisables. Afin de comprendre ces formats de données, voici les diagrammes de structure de données de chacun.

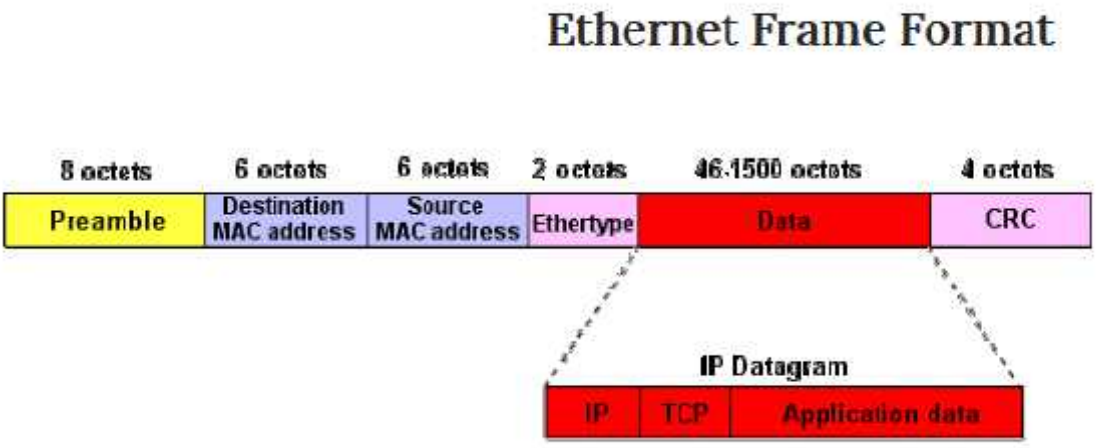


Figure 12 : Ethernet format.

Fondamentalement, pour extraire les données des paquets réseau, nous devons passer un argument qui représente les types de champs que nous voulons extraire dans la fonction << struct.unpack >>

Note :

b représente le type de données char

B représente le type de données char non signés

s représente le type de données string ("6s" = "ssssss»)

H représente les types de données short non signés

l représente le type de données long

L représente le type de données unsigned long

! Signifie extraire les données dans l'ordre inverse (le récepteur reçoit toujours les données dans l'ordre inverse pour diverses raisons liées à la mise en réseau)

Exemple :

Pattern = <SourceMac> + <DestinationMac> + <EthernetType>

Pattern = <"6s"> + <"6s"> + <"H">

Pattern = "! 6s6sH"

Voici le code qui nous permet d'extraire la trame Ethernet :

```python
# Ethernet Header
def eth_header(self, data):
 storeobj=data
 storeobj=struct.unpack("!6s6sH",storeobj)
 destination_mac=binascii.hexlify(storeobj[0])
 source_mac=binascii.hexlify(storeobj[1])
 eth_protocol=storeobj[2]
 data={"Destination Mac":destination_mac,
 "Source Mac":source_mac,
 "Protocol":eth_protocol}
 return data
```

Figure 13 : Trame Ethernet

Nous utilisons la même technique pour extraire (icmp, udp, tcp, ip) en utilisant ces diagrammes :

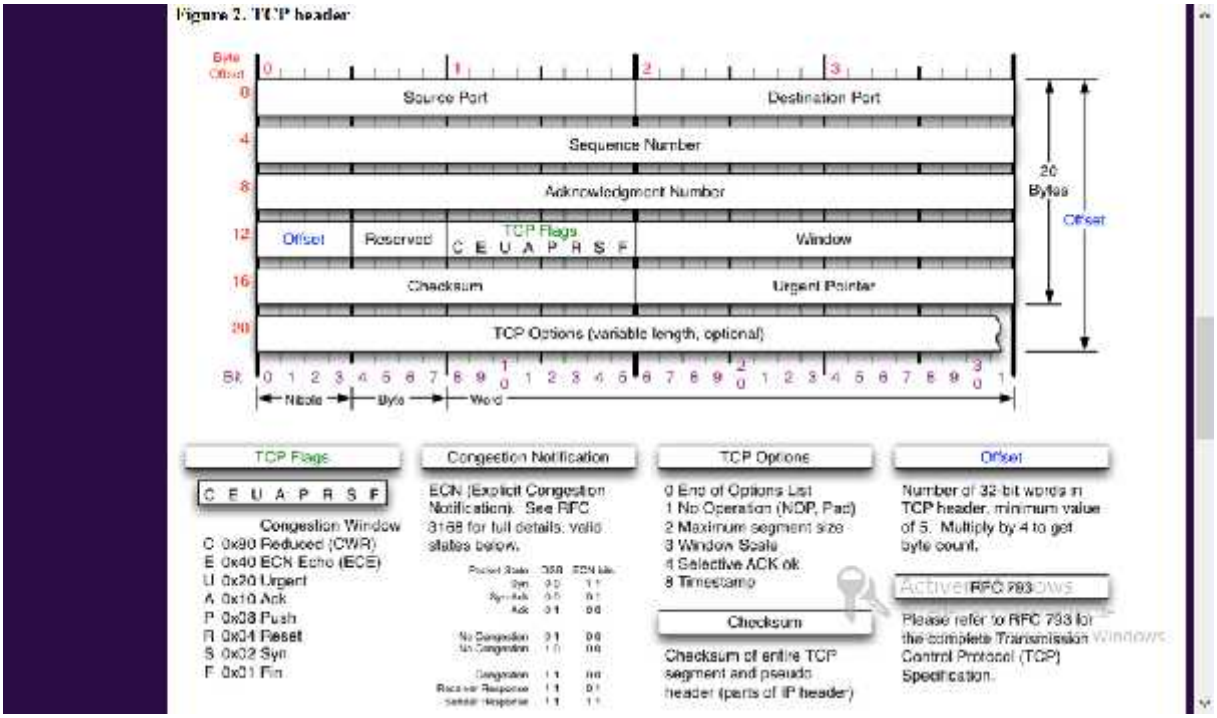


Figure 14 : TCP format.

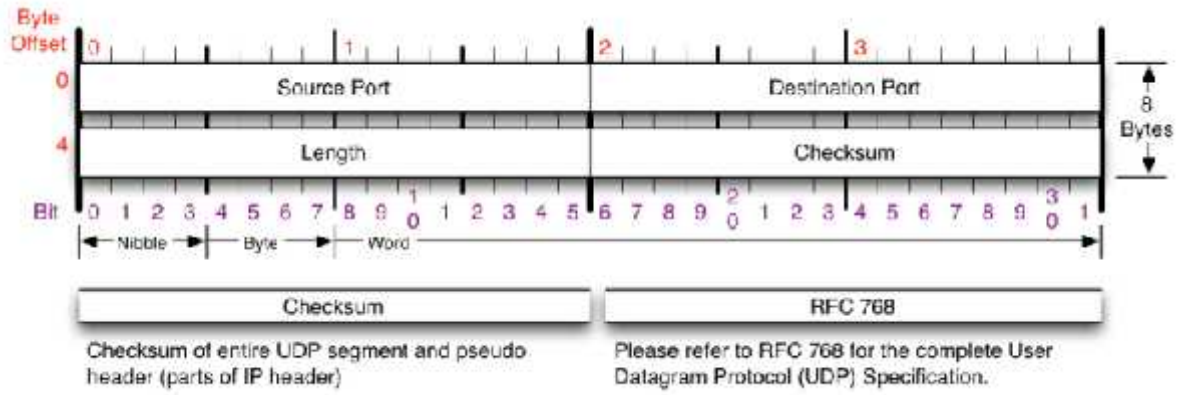


Figure 15 : UDP format.

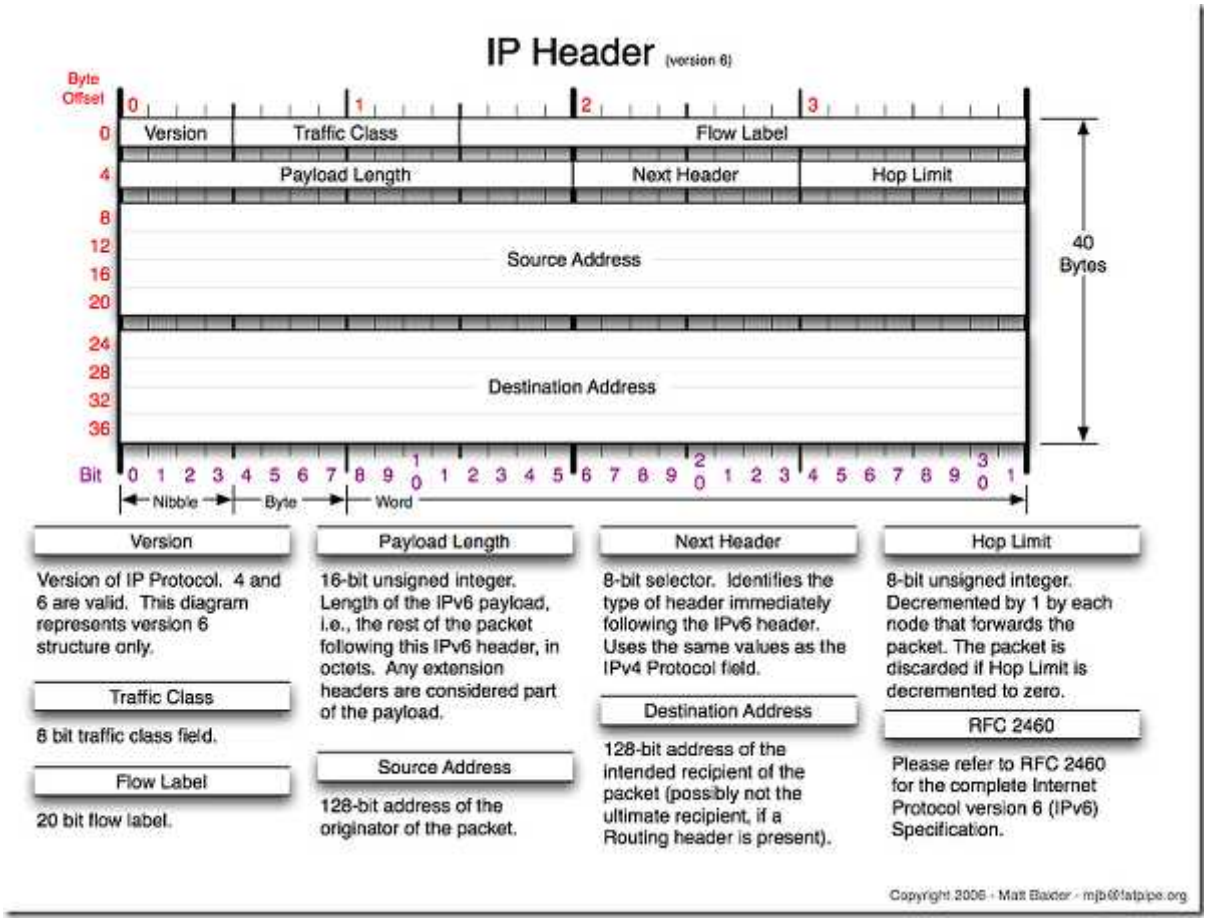


Figure 16 : IP format.

Voici ce qu'a intercepter l'Ids en ayant rafraichie la page Google :

Figure 17 : interception de donnée.

7. Chargement des données du dataset

Pour commencer, il faudra lire et charger les données contenues dans le fichier texte. Python propose via sa librairie Pandas des classes et fonctions pour lire divers formats de fichiers notamment les fichiers .Txt, La fonction *read_tablel ()*, renvoie un DataFrame contenant les 42 attributs du dataset

```
# Load NSL_KDD train dataset
dfkdd_train = pd.read_table("/home/rala/Documents/projet/NSL_KDD_dataset/KDDTrain.txt", sep=",", names=datacols)
```

Figure 18 : chargement des données.

Pour afficher les premières lignes (5 premières par exemple) chargées depuis notre fichier texte, on peut utiliser :

```
# View train data
print(dfkdd_train.head(5))
```

Figure 19 : affichage des 5 premières lignes du dataset Ttrain.

7.1 Normalisation des données et features Scaling

Notre exemple comporte des variables prédictives avec **des ordres de grandeurs très différents**.

Pour appliquer les algorithmes de classification, il est nécessaire que les variables prédictives faisant partie du modèle prédictif soient **du même ordre de grandeur.**

Pour ramener nos variables prédictives au même ordre de grandeur, nous appliquerons un procédé qui s’appelle : **features scaling**

La librairie Scikit learn de Python propose plusieurs classes et méthodes pour faire de la préparation de données (Data pré-processing) pour les algorithmes de Machine Learning. Le package sklearn.preprocessing propose la classe StandardScaler qui permettra de faire du features Scaling sur toutes nos variables prédictives.

Les données bien propres peuvent maintenant commencer à être explorées. Cette étape vous permet de mieux comprendre les différents comportements et de bien saisir le phénomène sous-jacent.

C'est vraiment une étape à ne pas négliger, les meilleurs data Scientists ne sont pas ceux qui connaissent les algorithmes les plus complexes mais ceux qui ont une très bonne connaissance des données, et ont préparé le terrain avec soin en amont.

7.2 Représentation des données

Pour mieux comprendre les données et voir quelques statistiques, il est souvent utile de les visualiser. On peut représenter les données dans un espace avec la librairie Matplotlib, voici un exemple de représentation graphique pour la fréquence d’apparition des attaques dans le dataset train et dans la base de test.

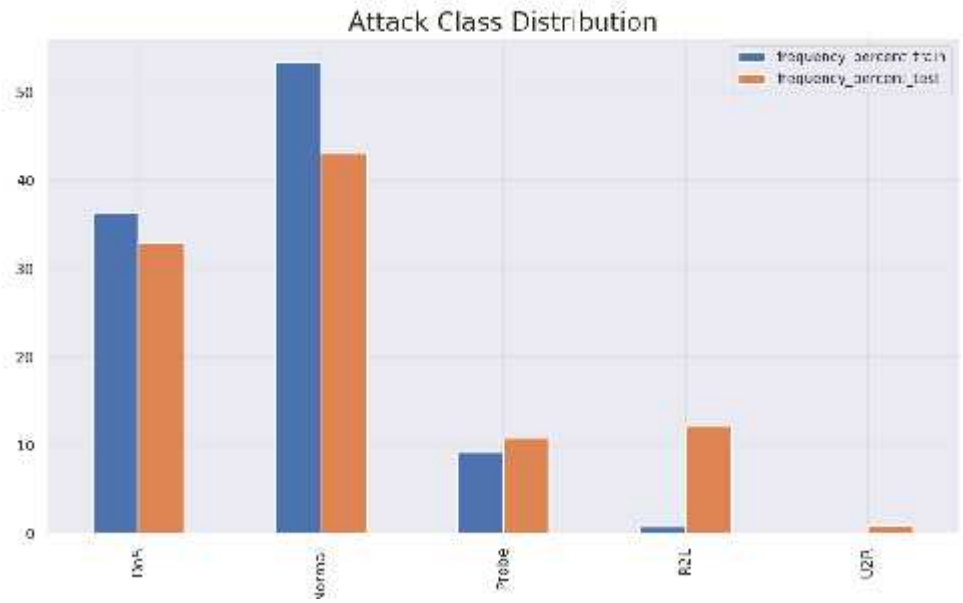


Figure 20 : fréquence d'apparition des classes.

8. Apprentissage de l’algorithme par les données

Nous allons enfin pouvoir appliquer nos algorithmes de classification, pour chaque algorithme nous allons montrer les différentes fonctions utilisés :

```
from sklearn.svm import SVC
from sklearn.naive_bayes import BernoulliNB
from sklearn import tree
from sklearn.model_selection import cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
```

Figure 21 : importation des algorithmes.

8.1 Machine à vecteur de support (svm) :

- Svc_classifier=svc(**random_state**=0)

Le germe du générateur de nombres pseudo aléatoires utilisé lors du brassage des données pour les estimations de probabilité. Random_state est le générateur de nombre aléatoire; S’il est nul, le générateur de nombres aléatoires est l'instance RandomState utilisée par np.random

```
# Train SVM Model
SVC_Classifier = SVC(random_state=0)
SVC_Classifier.fit(X_train, Y_train)
```

Figure 22 : algorithme SVM.

8.2 Naive Bayes:

- **From sklearn.naive_bayes import BernoulliNB**

Bernoulli NB implémente les algorithmes naïfs d’entraînement et de classification de Bayes pour les données distribuées selon des distributions multi variées de Bernoulli; c'est-à-dire qu'il peut y avoir plusieurs caractéristiques, mais chacune d'elles est supposée être une variable à valeur binaire (Bernoulli, booléen). Par conséquent, cette classe nécessite que les échantillons soient représentés en tant que vecteurs de caractéristiques à valeur binaire; si on lui transmet un autre type de données, une instance de Bernoulli NB peut bina riser son entrée (en fonction du paramètre binarize).

```
# Train Gaussian Naive Baye Model
BNB_Classifier = BernoulliNB()
BNB_Classifier.fit(X_train, Y_train)
```

Figure 23 : algorithme naïve bayes.

8.3 Knn:

- **From Sklearn .neighbors import KNeighborsClassifier**

scikit-learn implémente deux classificateurs différents voisins les plus proches: celui que nous utilisons est :

- KNeighborsClassifier : implémente l'apprentissage en fonction des voisins les plus proches de chaque point de requête, où k est une valeur entière spécifiée par l'utilisateur, Le choix optimal de la valeur dépend fortement des données: en général, une valeur supérieure supprime les effets du bruit, mais rend les limites de classification moins distinctes.
- n_jobs: Int ou None, facultatif (par défaut = None) : Nombre de travaux à exécuter en parallèle pour les ajustements et les prévisions. None signifie 1 sauf dans un contexte joblib.parallel_backend. -1 signifie utiliser tous les processeurs.

```
# Train KNeighborsClassifier Model
KNN_Classifier = KNeighborsClassifier(n_jobs=-1)
KNN_Classifier.fit(X_train, Y_train);
```

Figure 24 : algorithme KNN.

9. Métriques utilisés :

9.1 Cross validation : Le moyen le plus simple d'utiliser la validation croisée consiste à appeler la fonction d'assistance cross_val_score sur l'estimateur et le jeu de données.

```
scores = cross_val_score(v, X_train, Y_train, cv=10)
```

Figure 25 : méthode de cross-validation.

9.2 Accuracy :

sklearn.dcs.accuracy_score (*y_true, y_pred, normalize=True, sample_weight=none*)

Dans la classification multi-étiquettes, cette fonction calcule la précision des sous-ensembles: l'ensemble d'étiquettes prédites pour un échantillon doit correspondre exactement à l'ensemble d'étiquettes correspondant dans y_true.

9.3 Confusion matrix :

Par définition, une matrice de confusion est telle qu'elle est égale au nombre d'observations connues pour être dans le groupe mais prévues pour être dans le groupe.

Ainsi, dans la classification binaire, le nombre de vrais négatifs est C 0,0, les faux négatifs sont C1, 0, les vrais positifs C1, 1 et les faux positifs C0, 1.

9.4 Classification Report :

sklearn.metrics.classification_report: résumé textuel de la précision, rappel, score F1 pour chaque classe. Dictionnaire retourné si output_dict est true. Le dictionnaire a la structure suivante:

$$\text{Precision} = \frac{V \quad p}{(v \quad p \quad +f \quad p \quad)}$$

$$\text{Rappel} = \frac{V \quad p}{(v \quad p \quad +f \quad n \quad)}$$

$$\text{f-mesure} = \frac{2*p \quad *r}{p \quad +r}$$

```
from sklearn import metrics

models = []
models.append(('SVM Classifier', SVC_Classifier))
models.append(('KNeighborsClassifier', KNN_Classifier))
models.append(('Naive Baye Classifier', BNB_Classifier))
models.append(('Decision Tree Classifier', DTC_Classifier))
models.append(('RandomForest Classifier', RF_Classifier))
models.append(('LogisticRegression', LGR_Classifier))
```

Figure 26 : création d'une liste des model des classifieurs.

```
for i, v in models:
    scores = cross_val_score(v, X_train, Y_train, cv=10)
    accuracy = metrics.accuracy_score(Y_train, v.predict(X_train))
    confusion_matrix = metrics.confusion_matrix(Y_train, v.predict(X_train))
    classification = metrics.classification_report(Y_train, v.predict(X_train))
    print()
    print('============================== {} {} Model Evaluation ==============================' .format(grpclass, i))
    print()
    print ("Cross Validation Mean Score:" "\n", scores.mean())
    print()
    print ("Model Accuracy:" "\n", accuracy)
    print()
    print("Confusion matrix:" "\n", confusion_matrix)
    print()
    print("Classification report:" "\n", classification)
    print()
```

Figure 27 : calcul des métriques pour chaque model.

Chapitre 4 : Résultats et discussions

Sélection d'attributs :

Un choix pertinents des attributs impacte sur la qualité des résultats obtenues, la méthode de fouille de donnée sont plus efficaces s »il existe des connaissances sur les attributs de domaine, sur la priorité de ces attributs, sur les attributs moins important et les relations éventuelles entre les attributs.

Voici ce que nous a donné le diagramme en présentant le taux d'importance de chaque attribut

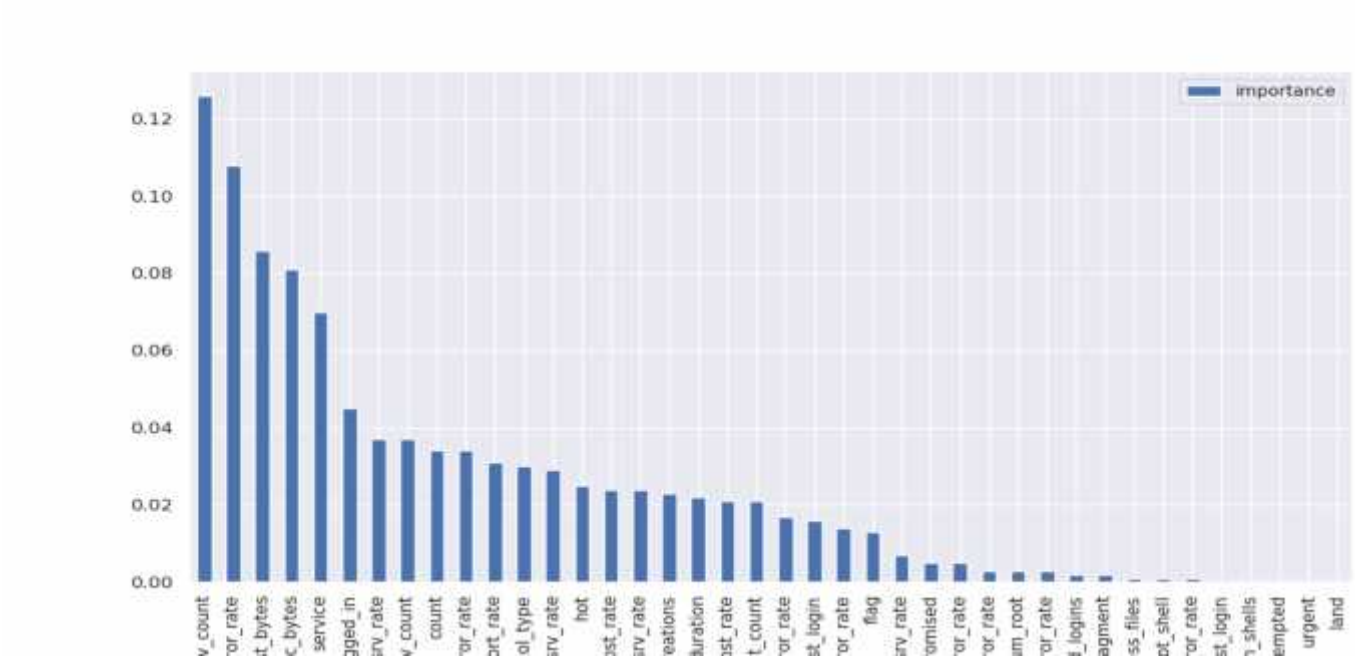


Figure 28 : taux d'importances de chaque attribut.

Nous avions pu tirer d'autre diagramme mais état donnée que chaque exécution nous prend 3H30, cela deviens impossible vu la grande capacité du dataset.

Résultats :

Résultats des métriques pour l'algorithme KNN

Figure 29 : métriques en utilisant l'algorithme KNN.

$$\begin{bmatrix} 67306 & 37 \\ 221 & 67122 \end{bmatrix}$$

True Negative (TN): C 0,0 = 67306

False Négative (FN) : C 1,0 = 221

Faux Positif (FP) : C 0,1 = 37

True Positive (TP) : C 1,1 = 67122

Résultats des métriques pour l'algorithme Naïve Bayes :

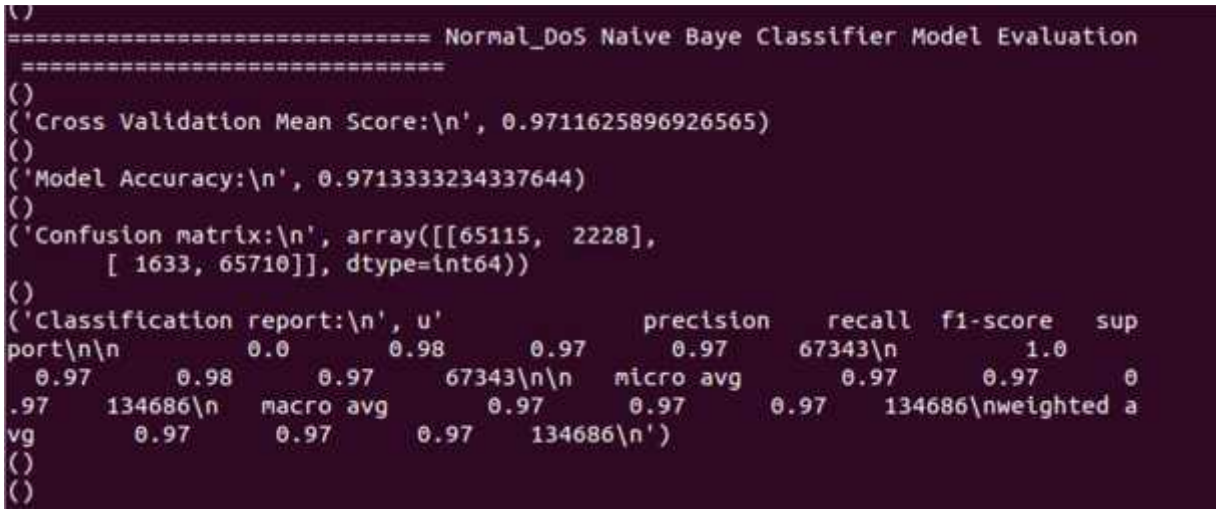


Figure 30 : résultats des métriques en utilisant Naïve Bayes.

Matrice de confusion

$$\begin{bmatrix} 65115 & 2228 \\ 1633 & 65710 \end{bmatrix}$$

True Negative (TN): C 0,0 = 65115

False Négative (FN) : C 1,0 = 1633

Faux Positif (FP) : C 0,1 = 2228

True Positive (TP) : C 1,1 = 65710

Résultats des métriques pour l'algorithme Décision tree :

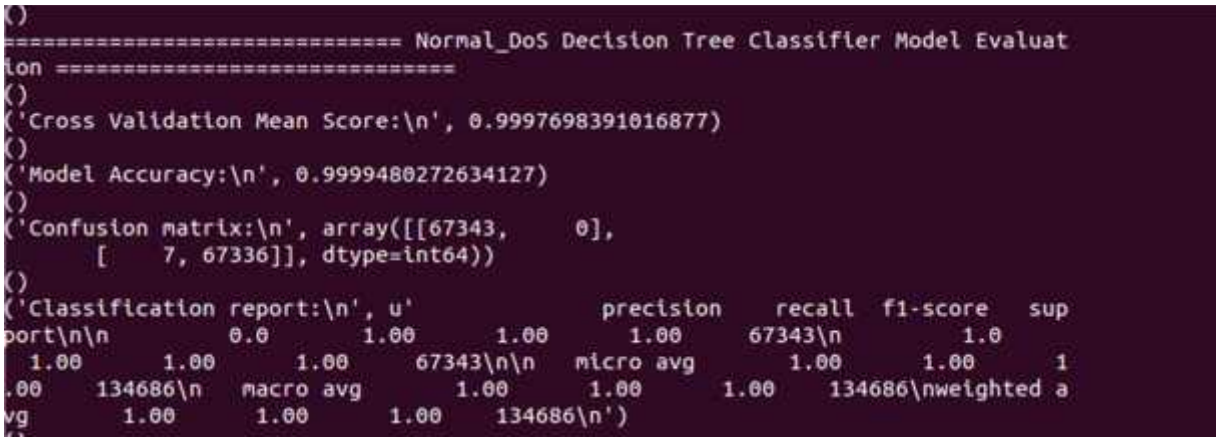


Figure 31 : résultats des métriques en utilisant les arbres de décisions.

$$\begin{bmatrix} 67343 & 0 \\ 7 & 67336 \end{bmatrix}$$

True Negative (TN): C 0,0 = 67343

False Négative (FN) : C 1,0 = 7

Faux Positif (FP) : C 0,1 = 0

True Positive (TP) : C 1,1 = 67336

Conclusion :

Nous avons présenté dans ce chapitre une solution pour la détection d’intrusion réseau se basant sur les algorithmes de la machine Learning, qui sont les algorithmes supervisés : KNN, Naïve Bayes, les arbres de décisions et les machines à vecteur de support. Les résultats d'analyse sur le jeu de données NSL-KDD montrent qu'il s'agit du meilleur jeu de données candidat pour simuler et tester les performances d'IDS. Cette analyse menée sur l'ensemble de données NSL-KDD à l'aide de figures et de tableaux aide le chercheur à bien comprendre l'ensemble de données. Il est également mis en évidence que la plupart des attaques sont lancées en utilisant les inconvénients inhérents au protocole TCP. Il est proposé d’examiner à l’avenir la possibilité de recourir à des techniques d’optimisation pour développer un modèle de détection d’intrusion offrant un meilleur taux de précision

Conclusion générale et perspectives :

Le travail, ainsi présenté dans ce mémoire est lié au domaine de la sécurité des réseaux plus précisément de la détection d’intrusions.

Nous avons proposé un nouveau modèle de système de détection d’intrusion réseau (NIDS) à base de signature fonctionnant sur la base des algorithmes de classification supervisée du machine Learning.

Nous nous sommes intéressés à la méthode des plus proches voisins Knn, où les expérimentations ont confirmé son succès déjà prouvé dans ce domaine.

Une autre méthode utilisant une approche différente pour la classification est investie. La méthode du bayes naïve, qui exploite la causalité pour réaliser l'inférence à la classe adéquate. Les probabilités sont mises en pratique, dans le but de couvrir la quantité énorme de données à étudier.

La décision de classification est plus rapide pour le bayes naïf et ses variantes, étant données qu'elle effectue une phase d'apprentissage et génère un modèle qu'elle utilise par la suite pour faire la classification. Cependant, le KNN consulte tout l'ensemble d'apprentissage à chaque nouveau processus à classifier.

Le KNN se montre robuste aux légers changements de l'ensemble des processus d'apprentissage, à l'égard de l'approche bayésienne qui est sensible au point que la structure. L'exactitude des deux approches est très satisfaisante surtout en ce qui concerne le taux de détection, même si nous constatons une performance visible du KNN qui a atteint un taux de détection maximal avec le plus petit faux positifs.

Les données de détection d'intrusions NSL KDD restent toujours le meilleur corpus de données libellées, qui permet aux concepteurs des IDS de construire, d'évaluer et de comparer les performances de leurs systèmes avec d'autres concepteurs qui ont utilisé le même ensemble de données.

Depuis, beaucoup de nouvelles technologies et de nouveaux protocoles ont été développés et beaucoup de nouveaux types d'attaques ont été détectés, d’où la nécessité d’enrichir cet ensemble de données.

D'autre part, les concepteurs des IDS doivent démontrer que leurs systèmes fonctionneront pour des ensembles de données autres que les données de NSL KDD.

Malheureusement, très peu de détails ont été publiés au sujet des techniques et des procédures sur les environnements de détection d'intrusions NSL KDD, la chose qui rend très difficile le fait de suivre les mêmes procédures pour refaire la collecte des données.

Bibliographie :

[1] Secure my data.*politique de securité informatique*. Disponible sur :

< https://www.securemydata.fr/formation/politique-de-securite-informatique/> (11 juillet 2019)

[2] CHIKOUCHE, S.Données d'apprentissage. In *Système de détection d'intrusion basé sur la classification comportementale des processus*. Université de M'sila:2011/2012,37

[3] TOUATI, A. La securité informatique. In *Détection d'intrusions dans les réseaux LAN : Installation et configuration de l'IDS-SNORT*. Université Bejaia : 2015/2016,1.

[4] JA-PSI. La politique de sécurité informatique. Disponible sur : < https://repo.zenk-security.com/Others/La%20politique%20de%20securite%20informatique.pdf > (11 juillet 2019) .PAGE 2

[5].Allam, K. and Hamidi, Z. La sécurité informatique. In *Meta-heuristique pour la détection d'intrusion : Application de la PSO*. Université Dr Tahar Moulay de Saida.2013/2014,3

[6] intrapole. (25 mars).Confidentialité / Intégrité / Disponibilité .Disponible sur :

< http://www.intrapole.com/spip.php?article18> (11 juillet 2019)

[7] Mémoire online. *ETUDE DETAILLEE DE DEPLOIEMENT*. Disponible sur :

< https://www.memoireonline.com/10/12/6259/m_Optimisation-de-la-securite-dans-un-environnement-de-travail-bancaire-cas-de-la-BSIC-Togo14.html > (11 juillet 2019)

[8] Moodle.Réseaux informatiques : Risques et enjeux .Disponible sur :

< https://moodle.utc.fr/pluginfile.php/16777/mod_resource/content/0/SupportIntroSecu/co/CoursSecurite_8.html>. (11 juillet 2019)

[9] Siener Informatique. *Prévention contre les vulnérabilités informatiques en respect de la RGPD*. Disponible sur :< http://blog.sienerinformatique.com/prevention-contre-vulnerabilites-informatiques-respect-de-grpd/> (02 mars 2019)

[10] Connaissance informatique. Types *de vulnérabilité informatique*. Disponible sur : <http://www.ordinateur.cc/D%C3%A9pannage/PC-D%C3%A9pannage/192056.html> (02 mars 2019)

[11] Mathgon.Les menaces informatiques. Disponible sur :

< http://www.mathgon.com/Cours/ESMISAB/CM4.pdf> (02 mars 2019).PAGE 1

[12] wikipedia.Attaque par déni de service.Disponible sur :

< https://fr.wikipedia.org/wiki/Attaque_par_d%C3%A9ni_de_service> (02 mars 2019)

[13] tpesecuriteinformatique.La sécurité informatique. Disponible sur :

< https://tpesecuriteinformatique.wordpress.com/les-differentes-attaques-informatiques/ > (02 mars 2019)

[14] SEHLA, Z. Sécurité des réseaux. Disponible sur :

< https://fr.slideshare.net/SehlaZayen/scurit-des-rseaux > .slide 15. (02 mars 2019)

[15] Coursehero.différents types d'attaques.Disponible sur

:< https://www.coursehero.com/file/p1ik7fg/6-II133-Diff%C3%A9rents-types-dattaques-Il-existe-deux-types-dattaques-a-Attaques/> (04 mars 2019)

[16] sécurité info. *Le DNS Spoofing : explications*. Disponible sur :

< https://www.securiteinfo.com/attaques/hacking/dnsspoofing.shtml > (06 mars 2019)

[17] g1site. *Méthode pour choisir un mot de passe sécurisé.* Disponible sur :

< https://www.g1site.com/methode-choisir-mots-de-passe/> (06 mars 2019)

[18] joomla comunity magazine.*comment-se-proteger-contre-les-attaques-par-force-brute.*Disponible sur :<https://magazine.joomla.org/international-stories-all/articles-in-french-all/comment-se-proteger-contre-les-attaques-par-force-brute> (07 mars 2019)

[19] wpserveur.*Une faille de sécurité XSS affecte de nombreux plugins.* Disponible sur : <https://www.wpserveur.net/une-faille-de-securite-xss-affecte-de-nombreux-plugins/> (07 mars 2019)

[20] astuce sécurité pc .*Sécurité informatique*. Disponible sur : < http://astuces-securite-pc-mac.blogspot.com/2008/03/les-attaques.html> (07 mars 2019)

[21] Wikipedia. *Exploit (informatique)* .Disponible sur :

< https://fr.wikipedia.org/wiki/Exploit_(informatique)> (08 mars 2019)

[22] Wikipedia. Shellcode. Disponible sur :< https://fr.wikipedia.org/wiki/Shellcode> (08 mars 2019)

[23] Wikipedia. *Dépassement de tampon*. Disponible sur : <https://fr.wikipedia.org/wiki/D%C3%A9passement_de_tampon >

[24] AKAOMA.*think security first* [en ligne]. Disponible sur : <https://www.akaoma.com/ressources/glossaire-securite-informatique/xss>. (11 juillet 2019)

[25] MABO, Ellie.Comment et avec quoi les systèmes sont-ils attaqués ? In *Cours sur l'introduction à l'informatique générale*.2010, 21

[26] HASSAIM, J. et HADDOUCHE, M. Introduction general. In *détection d'intrusion basée sur les réseaux AD HOC*.univ Bejaia : 2017/2018,8.

[27] Institut numérique. *Généralité sur les réseaux informatiques*[en ligne].Disponible sur :<https://www.institut-numerique.org/partie-1-generalites-sur-la-securite-des-reseaux-51dbb79863f85> (11 juillet 2019)

[28] Frédéric Majorczyk.Etat de l'art.In *Détection d'intrusions comportementale par diversification de COTS : application au cas des serveurs web.* Informatique [cs]. Université Rennes 1, 2008. Français. tel-00355366 ,9

[29] CHIKOUCHE, S. Introduction general. In *détection d'intrusion basée sur les réseaux AD HOC*.univ Msila : 2017/2018,8.

[30] TOUATI, A. Les systèmes de détection d'intrusion. In *Détection d'intrusions dans les réseaux LAN : Installation et configuration de l'IDS-SNORT*. Université Bejaia : 2015/2016,22.

[31] CHIKOUCHE, S.Généralité sur les systèmes de détection d'intrusion. *In Système de détection d'intrusion basé sur la classification comportementale des processus*. Université de M'sila:2011/2012,21

[32] IDS : Intrusion Détection Systems. Disponible sur :< ttp://www-igm.univ-mlv.fr/~dr/XPOSE2004/IDS/IDSSnort.html> (11 juillet 2019)

[33] mémoire online. *Mise en place d'un IDS en utilisant Snort*.Disponible sur : <https://www.memoireonline.com/04/15/9036/m_Mise-en-place-d-un-IDS-en-utilisant-Snort18.html> (11 juillet 2019)

[34] HADAOUI, R.Généralité sur les systèmes de détection d'intrusion.*In Système de détection d'intrusion basé sur la classification comportementale des processus*. Université M'Hamed BOUGARA de BOUMERDES:2008/2009,24

[35] SACI, S. et BATOUCHE, S. Principes de détection d'intrusion. In *<Etude et mise en place d'un système de Détection d'intrusion sous Linux>*.Université Abderrahmane Mira de Bejaïa:2014/2015.17

[36] digital insiders. Qu'est-ce que le Machine Learning .Disponible sur :

< https://digitalinsiders.feelandclic.com/construire/definition-quest-machine-learning >

(11 juillet 2019)

[37] e-marketing. Machine Learning. Disponible sur :

< https://www.e-marketing.fr/Definitions-Glossaire/Machine-learning-305604.htm >

(11 juillet 2019)

[38] Linked-in. 8 *Algorithmes de Machine learning expliqués en Langage Humain*. Disponible sur : < https://www.linkedin.com/pulse/8-algorithmes-de-machine-learning-expliqu%C3%A9s-en-humain-ga%C3%ABl> (11 juillet 2019)

[39] Mr. Mint. Machine *Learning made easy*. Disponible sur :< https://mrmint.fr/data-preprocessing-feature-scaling-python> (11 juillet 2019)

[40] Microsoft azure. Machine Learning. Disponible sur : < https://docs.microsoft.com/fr-fr/azure/machine-learning/team-data-science-process/prepare-data> (11 juillet 2019)

[41] Openclassroom.*Initiez vous au machine learning*. Disponible sur :

< https://openclassrooms.com/fr/courses/4011851-initiez-vous-au-machine-learning/4020631-exploitez-votre-jeu-de-donnees> (11 juillet 2019)

[42] LAROUMAGNE, F. *Le machine Learning automatisé*. Disponible sur : <https://jedha.co/blog/2018/10/17/le-machine-learning-automatise/> (11 juillet 2019).

[43] Openclassroom. evaluez et améliorez les performances d'un modèle de machine learning. Disponible sur : < https://openclassrooms.com/fr/courses/4297211-evaluez-et-ameliorez-les-performances-dun-modele-de-machine-learning> (11 juillet 2019)

[44] Mr. Mint. *Machine Learning made easy*. Disponible sur :< https://mrmint.fr/aborder-un-probleme-de-machine-learning-partie-2> (11 juillet 2019)

[45] Globb security. *Le Machine Learning en cybersécurité: back to basics*. Disponible sur :

< http://globbsecurity.fr/machine-learning-cybersecurite-back-to-basics-43596/> (11 juillet 2019)

[46] Studylib exploré. *Classification et apprentissage actif à partir d`un flux de données*. Disponible sur : <https://studylibfr.com/doc/2015890/classification-et-apprentissage-actif-%C3%A0-partir-d-un-flux-... > (11 juillet 2019)

[47] Xavier Dupre. Bien démarrer un projet de machine learning.Disponible sur : <https://studylibfr.com/doc/2015890/classification-et-apprentissage-actif-%C3%A0-partir-d-un-flux-... > (11 juillet 2019)

[48] Prevn. Apprentissage automatique (Machine Learning). Disponible sur : <> (11 juillet 2019)

[49] Mr mint. Machine Learning made easy. Disponible sur :< https://mrmint.fr/naive-bayes-classifier> (11 juillet 2019)

[50] Openclasseroom. Modelisez vos données avec les méthodes ensemblistes. Disponible sur :

<

https://openclassrooms.com/fr/courses/4470521-modelisez-vos-donnees-avec-les-methodes-ensemblistes/4664688-reduisez-la-correlation-entre-les-apprenants-faibles-a-l-aide-des-forets-aleatoires>. (11 juillet 2019)

[51] zeste de savoir. Un peu de machine Learning avec les svm. Disponible sur :

< https://zestedesavoir.com/tutoriels/1760/un-peu-de-machine-learning-avec-les-svm/> (11 juillet 2019)

[52] Mr. Mint. Machine Learning made easy. Disponible sur :< https://mrmint.fr/introduction-k-nearest-neighbors> (11 juillet 2019)