République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Chadli Bendjedid
Faculté des Sciences et de la Technologie
Département d'Informatique

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشاذلي بن جديد
كلية العلوم و التكنولوجيا
قسم الإعلام الآلي.

MEMOIRE

Présentée par

Melle Younes Ines

Pour l'obtention de diplôme de

MASTER
Filière : Informatique

Spécialité : Systèmes Informatiques Intelligents

Thème

Prédiction et détection d'anomalies Réseaux par Réseaux de neurones artificiels

Soutenue le : 2020 Devant le Jury composé de :

Qualité	Nom et Prénom	Grade	Université
Présidente	Mme. Nada Ahmedmalek	MAB	Chadli Benedidj.El Tarf
Rapporteur	**Mme. Zekri Meriem**	MCB	Chadli Benjedid. El Tarf
Examinateur	Dr. Fouzia Anguel	MCB	Chadli Benjedid.El Tarf

Année Universitaire : 2019/2020

Remerciements

Avant tout, nous remercions Dieu le tout puissant de nous avoir aidé à mener à bout ce travail, et à le concrétiser.

Nous remercions vivement Madame« Zekri Meriem» d'avoir accepté de diriger ce mémoire et pour ses conseils et orientations avisés.

Nous remercions vivement les membres du jury qui nous ont fait le grand honneur d'accepter de juger notre travail.

Notre reconnaissance s'adresse à tous les enseignants du département 'informatique pour l'enseignement et les conseils qu'ils nous ont apportés durant toute notre formation.

Ainsi que toutes les personnes qui, de pré ou de loin, ont généreusement contribués à l'élaboration de ce mémoire.

Dédicaces

Avant tout je rends grâce a dieu de m'avoir donne la force et le courage d'achever ce travail.

Je dédier ce modeste travail à :

Mes très chers père Younes et mère Fadda

Je ne trouverai jamais de mots pour vous exprimer mon profond attachement et ma reconnaissance pour Lamour, la tendresse et surtout pour votre présence dans mes moments les plus difficiles.

Vifs remercîment s'adressent à mes sœurs Dounia, Aya et mon frère Abdoul moula

A tous mes familles Nejet ,Jennet .

A tous les amies qui mon encouragés surtout AttawaAnissa ;

Et mes amis Semyaoui Fozia, Semyaoui somia, Ramdani youssria Shelghom Chahinez.

Que dieu leur accorde santé et prospérité.

Younes Ines

Table des matières

Table des figures

Introduction générale

La détection d'anomalies (outliers) est en enjeu ancien et majeur des applications industrielles de la Statistique notamment pour la détection d'une défaillance ou défaut de fabrication. Historiquement très présente dans les services de suivi de la qualité par contrôle statistique des procédés (Statistical Process Control), la détection d'anomalies utilise des techniques largement répandues et imposées, pour cela nous avons décidé de concevoir un système de détection d'anomalie par réseau de neurone.

1. Contexte du projet

Notre sujet s'intitule « Prédiction et détection d'anomalie par réseau de neurone», en effet nous voulons mettre en place un système de détection d'anomalie efficace de dut de minimiser les attaques contre les systèmes informatique tout en utilisant l'outil informatique pour permettre un bon système pour détecter les anomalies.

2. Problématique

Avant l'apparition des systèmes de détection d'anomalie, les systèmes informatiques ont été attaqués par des virus de toute sorte *où* les systèmes sont sabotés mais après la création des systèmes de détection d'intrusion tous les peuples travaillent allaise sans peur de virus *où autre attaques.* On minimisé ce problème et repousse fondamentalement des solutions :

- Travailler sur un PC protégé.
- une gestion automatique pour le système.

Cette application de détection d'anomalie par réseau de neurone composé en deux classe normale et anormale a partir les ce deux classe on peut organiser les donnée système.

3. L'objectif :

L'objectif de notre projet consiste à développer un système pour la détection d'anomalie par réseau de neurone, qui permettra d'offrir plusieurs services et de réaliser différentes opérations, citant à titre d'exemple

- Résoudre les problèmes d'anomalie dans les systèmes informatique.
- Protéger vos ordinateur (les fichiers, les dossiers….).
- repérer des données qui ne sont pas conformes à ce à quoi l'on peut s'attendre par rapport aux autres données.
- *Éliminer* les virus.

4. Contenu du mémoire

Le mémoire est organisé en trois chapitres comme suit :

- Dans le premier chapitre «La détection d'intrusion», on a présenté une vue générale sur les systèmes de détection d'intrusion(les approches, efficacité domaine appliquée, classification).
- Dans le deuxième chapitre «Machine Learning ou bien apprentissage automatique», on a détaillé les méthodes proposées d'apprentissage automatique en se basant sur réseau de neurone comme une méthode principale dans notre système.
- Dans le troisième chapitre intitulé «Implémentation et présentation de l'application», on a examiné tout d'abord les technologies et les outils utilisés dans la réalisation de notre projet. Puis, on a expliqué les différentes tâches et fonctionnalités assurées pour réaliser notre système à travers une série de captures réelles de l'interface du système et des différentes pages

Chapitre 1 : La détection d'intrusion

1. Introduction

Dans ce chapitre nous allons aborder les concepts du système de détection d'intrusion, qui consiste à détecter et analyser toute tentative d'effraction.

Nous commençons par introduire le domaine de la détection d'intrusion en générale. Ensuite, nous détaillerons les systèmes d'IDS.

Nous allons expliquer pourquoi les systèmes de détection d'intrusion sont nécessaires, on abordera également la classification des IDS, dans ce cadre plusieurs critères sont pris en compte.

Dans la deuxième section, notre système de détection d'intrusion basé sur différentes approches.

1.2. Introduction (du domaine)

De nos jours l'informatique et les réseaux ont envahi tous les domaines de la vie quotidienne, voire même qu'ils sont devenu indispensables. Ce développement est géré par des personnes spécialistes du domaine nommé informaticien. Entre ces derniers existe certains mal intentionnés qui travaillent jour et nuit an de nuire au bon fonctionnement de ces systèmes et d'obtenir des informations personnelles ou concernant la vie privée. Les cibles préférées sont les infrastructures gouvernementales et nationales. Ce milieu technologique a des vulnérabilités à des attaques et des données malicieuses courantes. Ces derniers ne cessent de se développer et d'évaluer du jour à l'autre ce qui rend la protection plus dur à y parvenir.

Certains développeurs on prévoyait et mise en œuvre des astuces et solutions afin de sécuriser les différents systèmes. Car l'évaluation des systèmes est déterminée à partir de sa fiabilité.

Alors la protection est un point majeur pour les administrateurs ce qui a mené à utiliser et de développer certains outils pour assurer cette dernière. Parmi les outils les plus fréquents on distingue:

- ✓ Les mécanismes de chiffrement pour assurer l'intégrité.
- ✓ Les pare-feu utilisé pour filtrer le trafic réseau.
- ✓ L'antivirus pour se protéger contre malveillant et les données malicieuses.
- ✓ Les détections d'intrusion pour détecter certains comportements illégaux.

Dans ce chapitre nous nous sommes intéressés à la sécurité informatique avec ses différents types ainsi qu'aux IDS avec leurs diverses méthodes qui permettent à mettre afin aux attaques [1].

1.3. Les systèmes de détection d'intrusions (IDS)

Ils représentent un ensemble de composants logiciels et matériels dont la fonction principale est de détecter et analyser toute tentative d'effraction (volontaire ou non).

Fonctionnalité

Pour bien gérer un système de détection d'intrusions, il est important de comprendre comment celui-ci fonctionne **:**

- ✓ Détection des techniques de sondage (balayages de ports, fingerprinting), des tentatives de compromission de systèmes, d'activités suspectes internes, des activités virales ou encore audit des fichiers de journaux (logs).

Certains termes sont souvent employés quand on parle d'IDS :

• **Faux positif** : une alerte provenant d'un IDS mais qui ne correspond pas à une attaque réelle.

• **Faux négatif** : une intrusion réelle qui n'a pas été détectée par l'IDS [2]

1.4. Comparaison entre les types d'IDS

Il existe plusieurs types d'IDS, mais ils peuvent être classés en trois familles et comparés en fonction de leurs inconvénients et avantage

	Avantages	Inconvenients
NIDS	-Les captures peuvent êtres bien sécurisés puisqu' ils se contentent d'observer le trafic. -Détecter plus facilement les scans grâce aux signatures. -Assurer la sécurité contre les attaques puisqu' il est invisible.	-La probabilité de faux négatifs attaques non détectées est élevée et il est difficile de contrôler le réseau entier -ils doivent principalement fonctionner de manière cryptée d'ou une complication de l'analyse des paquets -A l'opposé des IDS basses sur l'hôte ; ils ne voient pas les impacts dune attaque.
HIDS	-Découvrir plus facilement un Cheval de Troie puisque les informations et les possibilités sont très étendues. -Détecter des attaques Impossible à détecter avec des IDS réseau puisque le trafic est souvent crypte. -Observer les activités sur l`hôte avec précision.	-Ils ont moins de facilite a détecter les scans. -Ils sont plus vulnérables aux attaques de type DoS. -Ils consomment beaucoup de ressources CPU.
Hybrides	-Moins de faux positifs -Meilleure corrélation (la corrélation permet de générer de nouvelles alertes à partir de celles existantes). -Possibilité de réaction sur les analyseurs.	-Taux élevé de faux positifs.

Figure1.1.comparaison entre les types d'IDS

1.5. Efficacité des systèmes de détection d'intrusions

L'efficacité d'un système de détection d'intrusions est déterminée par les mesures suivantes : **Exactitude** : Le système de détection d'intrusions n'est pas exact s'il considère les actions légitimes des utilisateurs comme a typiques ou intrusives (faux positif). [3]

Performance : Effectuer une détection en temps réel.

Tolérance aux pannes : Un système de détection d'intrusions doit être résistant aux attaques. **Rapidité** : Un système de détection d'intrusions doit exécuter et propager son analyse d'une manière prompte pour permettre une réaction rapide dans le cas d'existence d'une attaque pour permettre à l'agent de sécurité de réagir. **Complétude** : La complétude est la capacité d'un système de détection d'intrusion de détecter toutes les attaques [4].

1.6. Les approches de détection d'intrusion

On distingue deux grands types d'approches pour détecter des intrusions. La première consiste à rechercher des signatures connues d'attaques tandis que la seconde consiste à définir un comportement normal du système et à rechercher ce qui ne correspond pas à ce comportement. Un système de détection d'intrusions par recherche de signatures connaît ce qui est mal, alors qu'un système de détection d'intrusions par analyse de comportement connaît ce qui est bien. On parle de détection de malveillances et de détection d'anomalies [5].

1.6.1 Détection de malveillances

La détection de malveillances fonctionne essentiellement par la recherche d'activités abusives par comparaison avec des descriptions abstraites de ce qui est considéré comme malveillant.

Cette approche tente de mettre en forme des règles qui décrivent les usages non désirés, en s'appuyant sur des intrusions passées ou des faiblesses théoriques connues. Les règles peuvent être faites pour reconnaître un seul événement qui est en lui-même dangereux pour le système ou une séquence d'événements représentant un scénario d'intrusion. L'efficacité de cette détection repose sur l'acuité et la couverture de tous les abus possibles par les règles.

Mise en œuvre :

Systèmes experts : ils peuvent être utilisés pour coder les signatures de malveillance avec des règles d'implication si . . . alors les signatures décrivent un aspect d'une attaque ou d'une classe d'attaque. Il est possible de rentrer de nouvelles règles pour détecter de nouvelles attaques. Les règles deviennent généralement très spécifiques au système cible et donc sont peu portables. Raisonnement sur des modèles : on essaye de modéliser les malveillances à un niveau élevé et intuitif d'abstraction en termes de séquences d'événements qui définissent l'intrusion. Cette technique peut être utile pour l'identification d'intrusions

qui sont proches mais différentes. Elle permet aussi décibler les données surs lesquelles une analyse approfondie doit être faite. Mais entant qu'approche recherchant des signatures, elle ne peut que trouver des attaques déjà connues. Analyse des transitions d'états : on crée un modèle tel qu'à l'état initial le système ne soit pas compromis. L'intrus accède au système. Il exécute une série d'actions qui provoquent des transitions sur les états du modèle, qui peuvent être des états ou l'on considère le système compromis. Cette approche de haut niveau peut reconnaître des variations d'attaques qui passeraient inaperçues avec des approches de plus bas niveau .USTAT (voir section 4.5) est une implémentation de cette technique. Réseaux neuronaux : la flexibilité apportée par les réseaux neuronaux peut permettre d'analyser des données même si elles sont incomplètes ou déformées. Ils peuvent de plus permettre une analyse non-linéaire de ces données. Leur rapidité permet l'analyse d'importants flux d'audit en temps réel. On peut utiliser les réseaux neuronaux pour filtrer et sélectionner les informations suspectes pour permettre une analyse détaillée par un système expert. On peut aussi les utiliser directement pour la détection de malveillances. Mais leur apprentissage est extrêmement délicat, est il est difficile de savoir quand un réseau est prêt pour l'utilisation. On peut également lui reprocher son côté boite noire (on ne peut pas interpréter les coefficients).

Algorithmes génétiques :on définit chaque scénario d'attaque comme un ensemble pas forcément ordonné d'événements .Lorsqu'on veut tenir compte de tous les entremêlements possibles entre ces ensembles, l'explosion combinatoire qui en résulte interdit l'usage d'algorithmes de recherche traditionnels, et les algorithmes génétiques sont d'un grand secours. La détection d'intrusion par recherche de scenarii repose sur une base design autres d'intrusions et recherche ces signatures dans le journal d'audits. On peut rapprocher les méthodes utilisées à celles que l'on peut rencontrer dans le domaine des antivirus, où on recherche la signature de certains programmes dans les fichiers du système informatique, ou encore dans le domaine de la génomique où l'on recherche une séquence d'ADN dans un brin, ou ,plus généralement, tout ce qui s'apparente à l'appariement de séquences.

1.6.2 Détection d'anomalies

Cette approche se base sur l'hypothèse que l'exploitation d'une faille du système nécessite une utilisation anormale de ce système, et donc un comportement in habituel de l'utilisateur.

Mise en œuvre

Il est cependant difficile de caractériser un comportement intrusif en termes de seuils, et on risque beaucoup de fausses alarmes ou beaucoup d'intrusions non détectées sur une population d'usagers non uniforme. Au fur et à mesure que l'utilisateur change ses activités, son profil de travail attendu se met à jour. Certains systèmes tentent de concilier l'utilisation de profils à court terme et de profils à long terme. Il reste cependant difficile de profiler un utilisateur irrégulier ou très dynamique. De plus, un utilisateur peut

arriver à habituer lentement le système à un comportement intrusif. Un profil de groupe est calculé en fonction de l'historique des activités du groupe entier. On vérifie que les individus du groupe travaillent de la manière que le groupe entier a définie par son profil. Il est de plus parfois nécessaire de créer un groupe pour un seul individu. On peut aussi observer les changements dans l'utilisation des protocoles réseau, rechercher les ports qui voient leur trafic augmenter anormalement. Ces profils ne dépendant pas des utilisateurs et peuvent permettre la détection de plusieurs intrus qui collaboreraient. Les écarts par rapport au profil sont cependant très durs à interpréter. Les virus, chevaux de Troie, vers, bombes logiques et autres programmes du même goût se voient démasqués en profilant la façon dont les objets du système comme les fichiers ou les imprimantes sont utilisés. Le profilage peut se faire par type d'exécutable. Profilage statique c'est un profilage où la mise à jour du profil ne se fait pas en permanence mais seulement de temps en temps, et avec la bénédiction de la personne chargée de la sécurité. Contrairement à la détection de malveillances à base de règles, on n'a pas besoin des connaissances d'un expert. Les règles d'utilisation normale son générées automatiquement pendant la période d'apprentissage. Ils peuvent offrir un modèle plus efficace et moins complexe que les moyennes et les déviations standard. L'utilisation de réseaux neuronaux doit encore faire ses preuves, et même s'ils peuvent s'avérer moins gourmands en ressources, une longue et minutieuse phase d'apprentissage est requise. Le système immunologique montre en effet beaucoup d'aspects intéressants comme son mode d'opération distribué qui lui permet de continuer à fonctionner même après des pertes, sa capacité à apprendre automatiquement de nouvelles attaques pour mieux réagir les prochaines fois qu'elles se présentent, sa capacité à détecter des attaques inconnues, etc. On peut voir l'approche immunologique de la détection d'anomalies comme une méthode de détection d'anomalie où l'on utilise les techniques de détection des malveillances. En effet, les techniques de détection d'anomalie connaissent ce qui est bien et vérifient en permanence que l'activité du système est normales, alors que les techniques de détection de malveillance connaissent ce qui est mal et sont à sa recherche. L'approche immunologique propose de rechercher ce qui est mal en connaissant ce qui est bien.

1.6.3 Systèmes hybrides

Pour tenter de compenser quelques inconvénients de chacune des techniques, certains systèmes utilisent une combinaison de la détection d'anomalies et de la détection de malveillances. Par exemple, un compte d'administrateur aura un profil qui lui permet d'accéder à certains fichiers sensibles, mais il peut être utile de vérifier que des attaques connues ne sont pas utilisées contre ces fichiers. À l'inverse, utiliser des fichiers comportant le mot "nucléaire" ne caractérise aucune signature d'attaque, mais il peut être intéressant de savoir que cela est arrivé si ce n'était pas dans les habitudes de l'utilisateur.

1.7. Domaines impliqués dans la détection d'intrusions

Data mining

Le but est ici d'extraire des modèles descriptifs des énormes volumes de données d'audit. Plusieurs algorithmes du domaine du data mining peuvent être utiles. Parmi ceux qu'on peut trouver dans [LSM] et utilisés par [LS98], [LSM99] ou [LNY_ 00], ou peut noter :

Classification : la classification associe chaque élément de donnée à une plusieurs catégories prédéfinies. Ces algorithmes de classification génèrent des classifier, sous forme d'arbres de décision ou de règles. Une application dans la détection d'intrusion serait d'appliquer ces algorithmes à une quantité suffisante de données d'audit normales ou anormales pour générer un classifié capable d'étiqueter comme appartenant à la catégorie normale ou anormale de nouvelles données d'audit.

Analyse de relations : on établit des relations entre différents champs d'éléments d'audit, comme par exemple la corrélation entre la commande et l'argument dans l'historique des commandes de l'interpréteur de commandes, pour construire des profils d'usage normal. Un programmeur, par exemple pourrait avoir une forte relation entre *emacs* et des fichiers *C*. On définit ainsi des règles d'association.

Analyse de séquences : Ces algorithmes tentent de découvrir quelles séquences temporelles d'événements se produisent souvent en même temps. On peut noter que [LNY_ 00] utilise le Common Intrusion Détection Framework (CIDF). Le CIDF est un effort pour développer des protocoles et des APIs pour permettre aux projets de recherche sur la détection d'intrusion de partager les informations et les ressources, et pour que les composants de détection d'intrusion puissent être réutilisés. Un Internet Engineering Task Force (IETF) working group a été creé et nommé Intrusion Detection Working Group (IDWG). Le CIDF est pour l'instant à l'état d'Internet Drafts qui sont en passe de devenir des RFC1

Système multi agent

Plusieurs domaines de recherche dans les Mobile Agents Intrusions Détection System (MAIDS) sont ouverts par [JMKM99]. Certains sont mis en oeuvre dans [BFFI_ 98],[HWHM98], [HWHM00] ou [JMKM00].

Détection multi-point La détection multi-point est faite en analysant les flux d'audit de plusieurs hôtes pour détecter des attaques distribuées ou autres stratégies d'attaques d'un réseau dans sa globalité. Il est rarement possible de transporter tous les flux d'audit à un IDS central, et même dans ce cas, la détection est difficile. Les agents mobiles peuvent apporter le calcul distribué, le fait qu'on transporte l'analyseur au flux d'audit et non le flux d'audit à l'analyseur. Les agents pourraient détecter ces attaques, les corréler, et découvrir les stratégies d'attaques distribuées.

Architecture résistante aux attaques Une architecture hiérarchique est souvent utilisée pour des raisons de performances et de centralisation du contrôle. Cette conception a souvent plusieurs lignes de

communication non-redondantes. Un intrus peut couper une branche de l'IDS, voire le désactiver totalement en le décapitant.

Les agents mobiles peuvent apporter quelques solutions à ce problème :

– une architecture complètement distribuée

– une architecture hiérarchique standard on un agent peut remplacer chaque nœud et ramener une fonctionnalité perdue

– des agents mobiles qui se déplacent quand une activité suspecte est détectée.

Partage de connaissances Souvent, plusieurs systèmes de détection d'intrusion différents fonctionnent sur un site. Idéalement, ils partageraient les informations sur les attaques récentes pour améliorer la détection d'attaques futures.

Même si ce n'est pas un domaine de recherche réservé aux MAIDS, ceux-ci permettent une approche plus naturelle de ce genre de choses.

Agents errants L'échantillonnage aléatoire est utilisé avec succès depuis de longues années dans le domaine du contrôle de qualité, et les mathématiques sous-jacentes sont bien comprises et les paramètres peuvent être calculés. Chaque agent effectue un test spécifique et peut errer aléatoirement d'hôte en hôte. Quand des tests indiquent la possibilité d'une intrusion, des tests plus poussés peuvent être effectués sur l'hôte suspect.

Imprévisibilité Un intrus peut pénétrer dans un système sans être immédiatement détecté par l'IDS. Cela peut lui laisser le temps d'effacer ses traces ou de neutraliser l'IDS.

Un MAIDS reste vulnérable à cela mais se trouve néanmoins pourvu de quelques avantages. Lorsqu'un agent arrive sur un hôte, il transporte du code non encore altéré, et pourrait par exemple tester l'intégrité de la plateforme IDS locale. L'arrivée des agents et leur manière de fonctionner peuvent être imprévisibles, et il peut être très dur de rester inaperçu.

Diversité génétique Diversité génétique Les IDS à base d'agents mobiles peuvent être vus comme un ensemble d'entités autonomes. Cependant, s'ils ne diffèrent que par leurs données, leurs tests peuvent devenir prévisibles. Une manière de faire les choses serait de décrire ce qu'il faut détecter dans un langage standard et de laisser chaque instance de la classe trouver une manière de le détecter.

Réseaux de neurones

Les réseaux neuronaux sont utilisés pour leur rapidité de traitement et leur relative résistance aux informations incomplètes ou déformées. [Can98] les utilise de deux manières différentes.

Ils sont d'abord utilisés comme filtres pour filtrer et sélectionner les parties suspectes dans les données d'audit. Celles-ci sont ensuite analysées par un système expert. On peut ainsi réduire les fausses alarmes, et on peut même augmenter la sensibilité du système expert car il ne travaille que sur des données suspectes.

Puis ils sont utilisés de façon à prendre seuls la décision de classer une séquence d'événements comme malveillante.

Immunologie

Le système immunitaire biologique est précisément conçu pour détecter et éliminer les infections. Le fait que le système immunitaire soit distribué, que plusieurs composants interagissent localement pour obtenir une protection globale, permet de ne pas avoir de centre de contrôle, de point faible. L'algorithme utilisé par le système immunitaire pour détecter les intrusions est appelé algorithme de sélection négative. Les lymphocytes sont appelés détecteurs négatifs parce qu'ils sont conçus pour se lier au non-soi, c'est-à-dire que lorsqu'un lymphocyte est activé, le système immunitaire répond à une intrusion. Les lymphocytes sont créés avec des récepteurs générés aléatoirement. Les lymphocytes qui arrivent à maturité sont donc sensés détecter le non-soi. Lorsque le système immunitaire rencontre pour la première fois un certain type d'agents pathogènes, il produit une réponse primaire, qui peut prendre plusieurs semaines pour éliminer l'infection. La réponse primaire est lente parce qu'il peut n'y avoir qu'un petit nombre de lymphocytes qui s'attachent à eux. Une fois l'infection éliminée, le système immunitaire garde cette population de lymphocytes qui avait une grande affinité avec ces agents pathogènes, alors que les autres lymphocytes ont une durée de vie de quelques jours. Les lymphocytes T sont créés et matures uniquement dans le thymus. Bien que l'on rencontre en ce lieu la grande majorité des protéines du corps, il se peut que certains lymphocytes arrivent à maturité avec un détecteur qui s'attache à des cellules saines. Ce signal provient du système immunitaire lui-même ou d'autres cellules.

[FH] ou [FHS97] tentent d'appliquer vaguement quelques un de ces principes. Par contre, ARTIS (ARTificial Immune System) [HF00] calque parfaitement tous ces comportements, plus finement détaillés dans [Clo] ou [HF00], pour arriver à un résultat très encourageant.

1.8. Classification des IDS :

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.

- ✓ L'emplacement d'IDS
- ✓ Les méthodes de détection
- ✓ Les types de réponse
- ✓ □ Fréquence d'utilisation

Nous présenterons d'abord la détection d'intrusion basée sur l'hôte, puis basée sur une application, avant de nous intéresser aux IDS réseaux (Network IDS : NIDS).

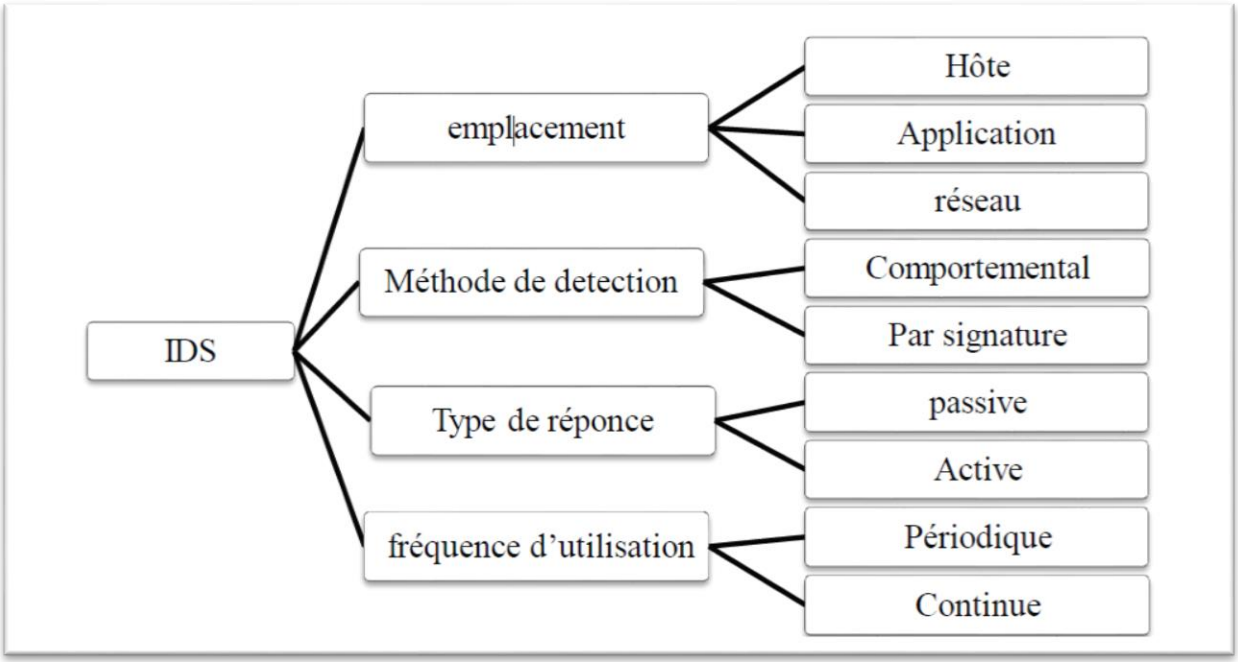

Figure. 1. 2. Les différentes sortes des IDS

La détection d'intrusion basée sur l'hôte :

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.

Un autre inconvénient tient justement à la taille des fichiers de rapport d'alertes à examiner, qui est très contraignante pour le responsable de sécurité. La taille des fichiers peut en effet atteindre plusieurs

Mégaoctets. Du fait de cette quantité de données à traiter, ils sont assez gourmands en CPU et peuvent parfois altérer les performances de la machine hôte.

Les HIDS sont en général placés sur des machines sensibles, susceptibles de subir des attaques et possédantes des données sensibles pour l'entreprise. Les serveurs web ou applicatifs, peuvent notamment être protégés par un HIDS.

La Détection d'Intrusion Réseau (NIDS)

La forme la plus commune de systèmes commerciaux de détection d'intrusion est basée sur le réseau. Ces systèmes détectent les attaques en capturant et en analysant les paquets réseau en écoutant sur le segment de réseau ou d'un commutateur. Ils le font en faisant correspondre entre un ou plusieurs paquets contre une base de données de signatures d'attaques connues, ou d'effectuer le décodage du protocole pour détecter les anomalies.

Le NIDS est capable à la fois de déclencher des alertes et de clôturer les connexions instantanément chaque fois qu'il remarque des activités suspectes. Le *« mode Promiscuous* » est la forme la plus commune du fonctionnement .ils surveillent chaque paquet qui est en transmission du segment local.

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. 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.

L'emplacement des capteurs :

Les capteurs placés sur le réseau sont placés en mode furtif (ou stealth mode), de façon à être invisibles aux autres machines. Pour cela, leur carte réseau est configurée en mode "promiscuous", c'est à dire le mode dans lequel la carte réseau lit l'ensemble du trafic, de plus aucune adresse IP n'est configurée. Un capteur possède en général deux cartes réseaux, une placée en mode furtif sur le réseau, l'autre permettant de le connecter à la console de sécurité.

Du fait de leur invisibilité sur le réseau, il est beaucoup plus difficile de les attaquer et de savoir qu'un IDS est utilisé sur ce réseau.

L'emplacement des capteurs :

Il est possible de placer les capteurs à différents endroits, en fonction de ce que l'on souhaite observer. Les capteurs peuvent être placés avant ou après le pare-feu, ou encore dans une zone sensible que l'on veut protéger spécialement.

Si les capteurs se trouvent après un pare-feu, il leur est plus facile de dire si le pare-feu a été mal configuré ou de savoir si une attaque est venue par ce pare-feu.

Ils ont pour mission de détecter les intrusions qui n'ont pas été arrêtées par ce dernier. Il s'agit d'une utilisation courante d'un NIDS.

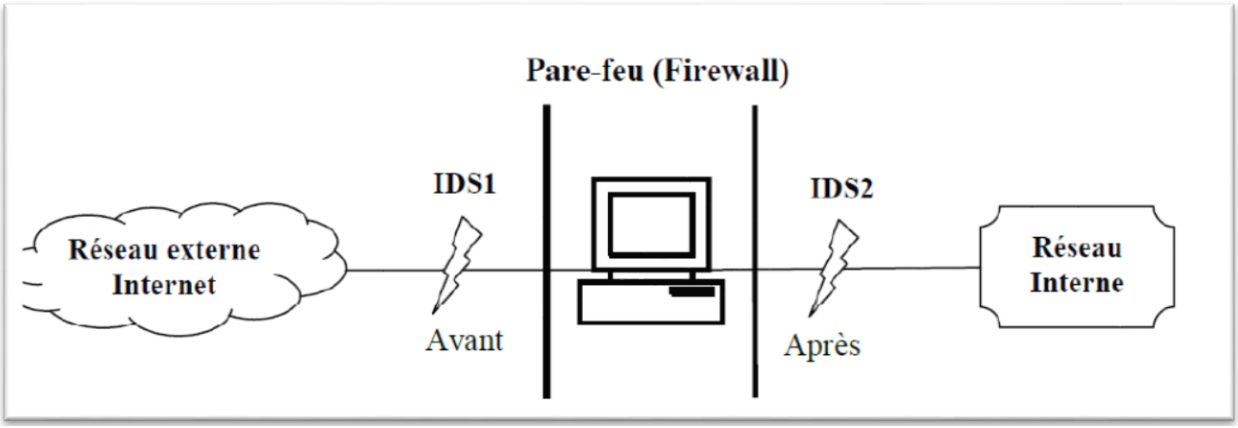

Figure. 1. 3. Installation des N-IDS

Il est également possible de placer un capteur à l'extérieur du pare-feu (avant le firewall). L'intérêt de cette position est que le capteur peut ainsi recevoir et analyser l'ensemble du trafic d'Internet. Si nous plaçons le capteur ici, il n'est pas certain que toutes les attaques soient filtrées et détectées. Pourtant, cet emplacement est le préféré de nombreux experts parce qu'il offre l'avantage d'écrire dans les logs et d'analyser les attaques (vers le pare-feu...), ainsi l'administrateur voit ce qu'il doit modifier dans la configuration du pare-feu.

Les capteurs placés à l'extérieur du pare-feu servent à détecter toutes les attaques en direction du réseau, leur tâche ici est donc plus de contrôler le fonctionnement et la configuration du firewall que d'assurer une protection contre toutes les intrusions détectées (certaines étant traitées par le firewall).

Il est également possible de placer un capteur et un autre après le firewall. En fait, cette variante réunit les deux cas mentionnés ci-dessus. Mais elle est très dangereuse si on configure mal les capteurs et/ou le pare-feu, en effet on ne peut simplement ajouter les avantages des deux cas précédents à cette variante.

Les capteurs IDS sont parfois situés à l'entrée de zones du réseau particulièrement sensibles (parcs de serveurs, données confidentielles…), de façon à surveiller tout trafic en direction de cette zone.

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. Son déploiement à peu d'impact sur le réseau, les NIDS sont généralement des dispositifs passifs qui écoutent sur le fil réseau sans interférer avec le fonctionnement normal d'un réseau. Un large réseau peut être contrôlé par quelques NIDS bien placés.

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) est élevée. Certains NIDS ont des difficultés à traiter des fragments de paquets, ce qui peut provoquer le dysfonctionnement d'un IDS. Ils ont des problèmes dans le traitement des vitesses élevées et des volumes élevés de trafic. Ils ne peuvent pas analyser les informations cryptées.

Pour finir, à l'opposé des IDS basés sur l'hôte, ils ne perçoivent pas les impacts d'une attaque .Même si nous distinguons HIDS et NIDS, la différence devient de plus en plus réduite puisque les HIDS possèdent maintenant les fonctionnalités de base des NIDS.

1.9. Conclusion

Dans ce chapitre, nous avons abordé les notions générales des systèmes de détection classification d'intrusions, leurs fonctionnements, ainsi que leurs fonctionnements. Ils complètent les taches des autres équipements de sécurité comme les par feux et VPN, anti-virus ...etc. Plusieurs méthodes sont été proposes pour réaliser les systèmes de détection d'intrusion comme réseaux de neurones

Chapitre 2 : L'apprentissage automatique

2.1. Introduction

Dans ce chapitre, nous essayerons, de parler sur l'apprentissage automatique la première partie est consacrée une vision globale sur l'apprentissage automatique (le définir, son historique, ses domaines d'application).

Dans la deuxième partie, on discute sur les types d'apprentissage (Supervisé, Semi_ supervisé, Non-supervisé), de façon plus détailler.

Dans la dernier partie, on fait une recherche sur Les méthodes du machine Learning cette partie présente les cinq méthodes (arbre de décision, Les machines à vecteurs de support (SVM), Les réseaux bayésiens, Les réseaux de neurones artificiels (RNA), K-means).

2.2. Introduction aux L'apprentissage automatique

L'apprentissage automatique est un sous-domaine de l'intelligence artificielle (IA). En général, l'objectif de l'apprentissage automatique est de comprendre la structure des données et de les intégrer dans des modèles qui peuvent être compris et utilisés par les tout le monde.

Bien que l'apprentissage automatique soit un domaine de l'informatique, il diffère des approches informatiques traditionnelles. En effet dans cette dernière, les algorithmes sont des ensembles d'instructions explicitement programmées utilisées par les ordinateurs pour calculer ou résoudre des problèmes. Les algorithmes d'apprentissage automatique permettent aux ordinateurs de s'entraîner sur les entrées de données et utilisent l'analyse statistique pour produire des valeurs qui se situent dans une plage spécifique. Pour cette raison, l'apprentissage automatique facilite l'utilisation des ordinateurs dans la construction de modèles à partir de données d'échantillonnage afin d'automatiser les processus de prise de décision en fonction des données saisies.

Tout utilisateur de la plus récente technologie bénéficie de l'apprentissage automatique. La technologie de reconnaissance faciale par exemple permet aux plateformes de médias sociaux d'aider les utilisateurs à marquer et partager des photos d'amis. La technologie de reconnaissance optique des caractères (OCR) convertit les images du texte en caractères mobiles. Les moteurs de recommandation, alimentés par l'apprentissage automatique, suggèrent les films ou émissions de télévision à regarder en fonction des préférences de l'utilisateur. Les voitures autonomes qui utiliseront l'apprentissage automatique pour naviguer seront bientôt disponibles pour les consommateurs.

L'apprentissage automatique est un domaine en développement continu. Pour cette raison, vous devez tenir compte de certaines considérations lorsque vous travaillez avec des technologies d'apprentissage automatique ou analysez l'impact des processus d'apprentissage automatique. Dans cet article, nous

étudierons les méthodes d'apprentissage supervisé et non supervisé, ainsi que les approches algorithmiques courantes de l'apprentissage automatique, y compris l'algorithme k plus proche voisin de l'anglais "**k-nearest neighbor**" ou **KNN**, l'apprentissage par arbre décisionnel et l'apprentissage en profondeur. Nous allons explorer quels langages de programmation sont les plus utilisés dans l'apprentissage automatique, en vous fournissant certains des attributs positifs et négatifs de chacun. De plus, nous discuterons des biais qui sont perpétués par les algorithmes d'apprentissage automatique, et considérons ce qui peut être gardé à l'esprit pour les éviter lors de la construction d'algorithmes [6].

2.3 Les types d'apprentissage automatique

Dans l'apprentissage automatique, les tâches sont généralement classées en grandes catégories. Ces catégories sont basées sur la façon dont l'apprentissage est reçu ou comment le feedback sur l'apprentissage est donné au système développé.

Deux des méthodes d'apprentissage automatique les plus largement adoptées sont **l' apprentissage supervisé** qui forme des algorithmes basés sur des données d'entrée et de sortie étiquetées par l'homme et **l'apprentissage non supervisé** qui ne fournit pas à l'algorithme des données étiquetées pour lui permettre de trouver une structure et de découvrir une logique dans données entrées. Explorons donc ces méthodes plus en détail.

- **L'apprentissage supervisée**

Dans l'apprentissage supervisé, l'ordinateur est fourni avec des exemples d'entrées qui sont étiquetés avec les sorties souhaitées. Le but de cette méthode est que l'algorithme puisse «apprendre» en comparant sa sortie réelle avec les sorties «enseignées» pour trouver des erreurs et modifier le modèle en conséquence. L'apprentissage supervisé utilise donc des modèles pour prédire les valeurs d'étiquettes sur des données non étiquetées supplémentaires.

Par exemple, avec un apprentissage supervisé, un algorithme peut être alimenté avec des images de requins étiquetés **Poisson** des images d'océans étiquetés comme **Ocean**. En étant formé sur ces données, l'algorithme d'apprentissage supervisé devrait être capable d'identifier plus tard des images de requin non marquées comme Poisson des images océaniques non étiquetées **Ocean**. Un cas d'utilisation de l'apprentissage supervisé consiste à utiliser des données historiques pour prédire des événements futurs statistiquement probables. Il peut utiliser les informations historiques sur les marchés boursiers pour anticiper les fluctuations à venir ou être utilisé pour filtrer les courriers indésirables. Dans l'apprentissage supervisé, des photos étiquetées de chiens peuvent être utilisées comme données d'entrée pour classer les photos non marquées de chiens.

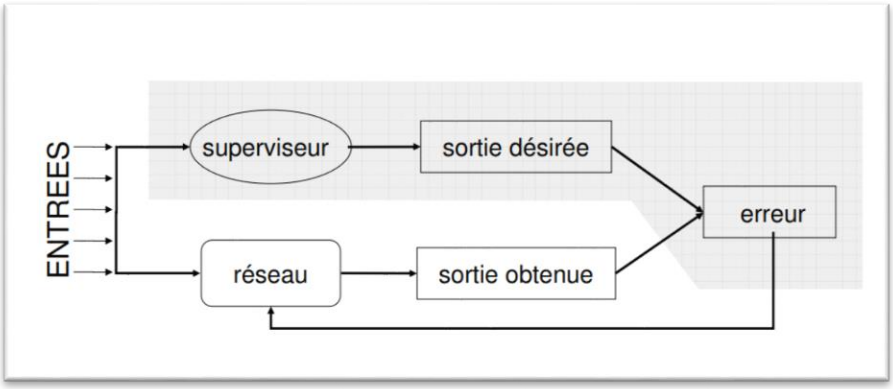

Figure2. 1 : apprentissage supervisé

- **L'apprentissage non supervisé**

Dans l'apprentissage non supervisé, les données sont non étiquetées, de sorte que l'algorithme d'apprentissage trouve tout seul des points communs parmi ses données d'entrée. Les données non étiquetées étant plus abondantes que les données étiquetées, les méthodes d'apprentissage automatique qui facilitent l'apprentissage non supervisé sont particulièrement utiles.

L'objectif de l'apprentissage non supervisé peut être aussi simple que de découvrir des modèles cachés dans un ensemble de données, mais il peut aussi avoir un objectif d'apprentissage des caractéristiques, qui permet à la machine intelligente de découvrir automatiquement les représentations nécessaires pour classer les données brutes.

L'apprentissage non supervisé est couramment utilisé pour les données transactionnelles. Vous pouvez avoir un grand ensemble de données sur les clients et leurs achats, mais en tant qu'être humain, vous ne serez probablement pas en mesure de comprendre quels attributs similaires peuvent être tirés des profils de clients et de leurs types d'achats. Avec ces données introduites dans un algorithme d'apprentissage non supervisé, on peut déterminer que les femmes d'une certaine tranche d'âge qui achètent des savons non parfumés sont susceptibles d'être enceintes, et donc une campagne de marketing liée à la grossesse et aux produits pour bébés pour augmenter leur nombre d'achats.

Sans une réponse «correcte», les méthodes d'apprentissage non supervisées peuvent examiner des données complexes, plus expansives et apparemment sans point commun, afin de les organiser de manière potentiellement significative. L'apprentissage non supervisé est souvent utilisé pour la détection d'anomalies, y compris pour les achats frauduleux de cartes de crédit et les systèmes de recommandation qui conseille sur les produits à acheter ensuite. Dans l'apprentissage non supervisé, les photos non marquées des chiens peuvent être utilisées comme données d'entrée pour l'algorithme afin de trouver des similitudes et de classer les photos de chiens ensemble [7].

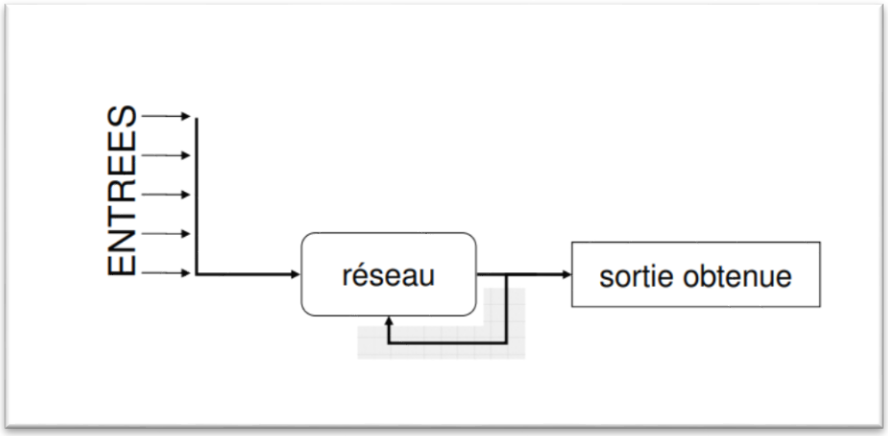

Figure 2.2 : apprentissage non supervisé

2.4 Les méthodes de l'apprentissage automatique

- **Arbre de décision**

Pour un usage général, les arbres de décision sont utilisés pour représenter visuellement les décisions et montrer ou éclairer la prise de décision. Lorsque vous travaillez avec l'apprentissage automatique et l'exploration de données, les arbres de décision sont utilisés comme modèle prédictif. Ces modèles cartographient les observations de données et tirent des conclusions sur la valeur cible des données.

L'objectif de l'apprentissage par arbre de décision est de créer un modèle qui prédira la valeur d'une cible en fonction de variables d'entrée. Dans le modèle prédictif, les attributs des données qui sont déterminés par l'observation sont représentés par les branches, tandis que les conclusions sur la valeur cible des données sont représentées dans les feuilles. Lors de l'apprentissage d'un arbre, les données source sont divisées en sous-ensembles en fonction d'un test de valeur d'attribut, qui est répété récursivement sur chacun des sous-ensembles dérivés. Une fois que le sous-ensemble d'un nœud a la valeur équivalente à sa valeur cible, le processus de récusions sera terminé. Regardons un exemple de diverses conditions qui peuvent déterminer si quelqu'un devrait lire à l'extérieur. Cela inclut les conditions météorologiques ainsi que les conditions de pression barométrique.

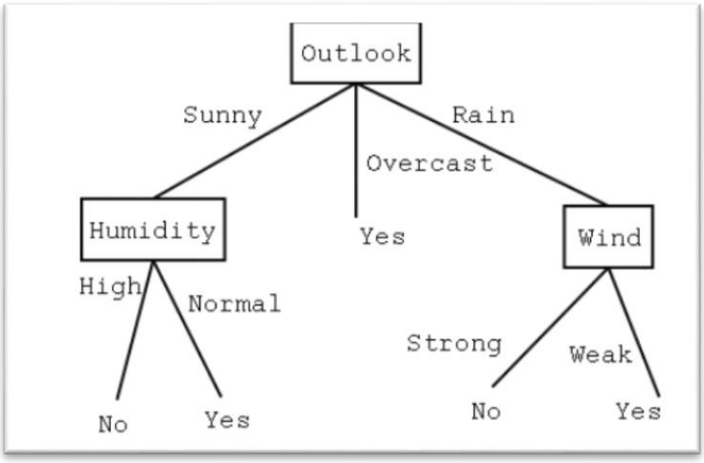

Figure 2.3 : arbre de décision

Dans l'arbre de décision simplifié ci-dessus, un exemple est classé en le triant à travers l'arbre vers le nœud feuille approprié. Ceci renvoie alors la classification associée à la feuille particulière, qui dans ce cas est soit a **Yes** ou **No**. L'arbre classe les conditions d'un jour en fonction de son aptitude ou non à se prêter à une lecture à l'extérieur. Un véritable jeu de données d'arbre de classification aurait beaucoup plus de fonctionnalités que ce qui est décrit ci-dessus, mais les relations devraient être simples à déterminer. Lorsque vous travaillez avec l'arbre de décision, plusieurs déterminations doivent être faites, y compris les caractéristiques à choisir, les conditions à utiliser pour la division et la compréhension lorsque l'arbre de décision a atteint une fin claire [8].

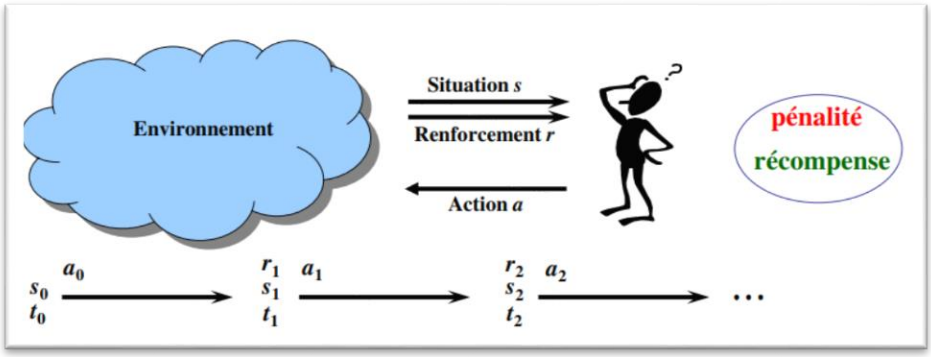

Figure2. 4 : Apprentissage par renforcement

- **Les machines à vecteurs de support (SVM)**

Parmi les méthodes a noyaux, inspirées de la théorie statistique de l'apprentissage de Vladimir Vapnik, les SVM constituent la forme la plus connue. SVM est une méthode de classification binaire par apprentissage supervise, elle fut introduite par Vapnik en 1995.

Cette méthode est donc une alternative récente pour la classification. Cette méthode repose sur l'existence d'un classificateur linéaire dans un espace approprie. Puisque c'est un problème de classification a deux classes, cette méthode fait appel a un jeu de données d'apprentissage pour apprendre les paramètres du modèle. Elle est basée sur l'utilisation de fonction dites noyau (kernel) qui permettent une séparation optimale des données.

Dans la présentation des principes de fonctionnements, nous schématiserons les données par des ≪ points ≫ dans un plan. La notion d'apprentissage étant importante, nous allons commencer par effectuer un rappel. L'apprentissage par induction permet d'arriver a des conclusions par l'examen d'exemples particuliers. Il se divise en apprentissage supervise et non supervise.

Le cas qui concerne les SVM est l'apprentissage supervise. Les exemples particuliers sont représentes par un ensemble de couples d'entrée/sortie. Le but est d'apprendre une fonction qui correspond aux exemples vus et qui prédit les sorties pour les entrées qui n'ont pas encore été vues.

Les entrées peuvent être des descriptions d'objets et les sorties la classe des objets donnes en entrée [9]

- **SVM principe de fonctionnement général**

- ✓ Notions de base: Hyperplan, marge et support vecteur Il existe une multitude d hyperplans valides Propriété des SVM : cet hyperplan doit être optimal celui qui passe «au milieu» des points des deux classes d exemples (= hyperplan le «plus sur»). Notions de base : Hyperplan, marge et support vecteur hyperplan dont la distance minimale aux exemples d apprentissage est maximale On appelle cette distance «marge» entre l hyperplan et les exemples L hyperplan séparateur optimal est celui qui maximise la marge. Comme on cherche a maximiser cette marge, on parlera de séparateurs à vaste marge. Notions de base : Hyperplan, marge et support vecteur Pourquoi maximiser la marge? Marge plus large => plus de sécurité lorsque l'on classe un nouvel exemple Idée: se comporte le mieux sur des données d'apprentissage => meilleur classement des nouveaux exemples Classification d un nouvel exemple (classe inconnue) Linéarité et non-linéarité modèles des SVM : cas linéairement séparable (classificateur linéaire facile à trouver) cas non linéairement séparable (la plupart des pb réels) **[10]**
- ✓ Cas non linéaire Idée des SVM : changer l espace des données séparation linéaire des exemples dans un nouvel espace changement de dimension Probabilité plus élevée de trouver un hyperplan séparateur Nouvelle dimension est appelée espace de re-description Exemple de transformation de cas non linéaire : le cas XOR cas de XOR: non linéairement séparable, Coordonnées des points : (0,0) ; (0,1) ; (1,0) ; (1,1) fonction polynomiale (x, y) (x, y, x.y) qui fait passer d'un espace de dimension 2 a un espace de dimension 3 problème en 3D linéairement séparable : (0,0) (0,0,0) (0,1) (0,1,0) (1,0) (1,0,0) (1,1) (1,1,1) Illustration de transformation de cas non linéaire : le cas XOR Cas non linéaire = Transformation d un problème de séparation non linéaire dans l espace de représentation en un problème de séparation linéaire dans un espace de re-description de plus grande dimension Transformation non linéaire réalisée via une fonction noyau Quelques familles de fonctions noyau paramétrables sont connues il revient a l utilisateur de SVM d effectuer des tests pour déterminer celle qui convient le mieux pour son application. Exemples de noyaux :

polynomial, gaussien, sigmoïde et laplacien 2. Fondements mathématiques Problème d'apprentissage phénomène f (éventuellement non déterministe) qui, a partir d'un certain jeu d'entrées x, produit une sortie y = f(x). But : retrouver cette fonction f a partir de la seule observation d'un certain nombre de couples entrée-sortie{(xi; yi) : i = 1,.., n} afin de «prédire» d autres événements Soit un couple (X, Y) de variables aléatoires a valeurs dans X x Y. Seul le cas Y = {-1, 1} (classification) nous intéresse ici (on peut facilement étendre au cas card(y) = m > 2 et au cas Y = R). La distribution jointe de (X, Y) est inconnue. Sachant qu on observe un échantillon S = {(X1, Y1),...,(Xn, Yn)} de n copies indépendantes de (X, Y), on veut: construire une fonction h : X Y telle que P(h(X)!= Y) soit minimale**[11].**

- **Les réseaux bayésiens**

- **Introduction**

Un modèle graphique est une famille de distribution de probabilités définie en termes de graphe orienté ou non. Il est constitué de nœuds qui représentent des variables aléatoires et des arcs représentant les relations de dépendances entre ces variables.

On distingue deux formes de modèles graphiques probabilistes : les modèles orientés et les modèles non orientés, basés respectivement sur les graphes acycliques orientés connus sous le nom de réseaux bayésiens (RB) [12] et les graphes non orientés , exemple les champs de Markov (HMM)[9] pour ces modèles graphiques probabilistes, non seulement ils bénéficient des avantages des modèles probabilistes, mais aussi ils représentent les avantages liés à leur représentation graphique.

En effet ils permettent de visualiser la structure et les propriétés de dépendance conditionnelles du modèle probabiliste correspondant. Ainsi, les RB sont une union entre la théorie des probabilités et la théorie des graphes.

- **Différents modèles graphiques des réseaux bayésiens**

Il existe plusieurs variantes des RB telles que [13] : les RB multi agents, les RB de niveaux deux, les RB orientés objets, les diagrammes d'influence, les RB dynamiques [14] (temporels), les RB multi entités, les filtres bayésiens [15]: qui sont des RB dynamiques particuliers et les RB adaptés à la classification tels que ; les RB naïf, les RB naïf augmenté, etc. .

En classification, particulièrement, les RB sont largement. Dans ce cas, le nœud parent est considéré comme une variable non observée précisant à quelle classe appartient chaque objet alors que les nœuds enfants sont des variables observées correspondant aux différents attributs caractérisant cet objet.

Plusieurs modèles sont conçus dans ce but. Parmi ces réseaux, on peut citer le RB naïf qui est le plus simple, le réseau bayésien augmenté par n'importe qu'elle structure ou par une structure arborescente et autres.

Les RB ont une structure simple et unique qui comprend deux niveaux. Le premier niveau contient un seul nœud parent et le second plusieurs enfants avec la forte hypothèse naïve d'indépendance conditionnelle des enfants (X) conditionnellement au parent. Ils sont largement utilisés pour résoudre des problèmes de classification [16]. La figure.2.5 rappelle le principe de fonctionnement et de classification par RBN.

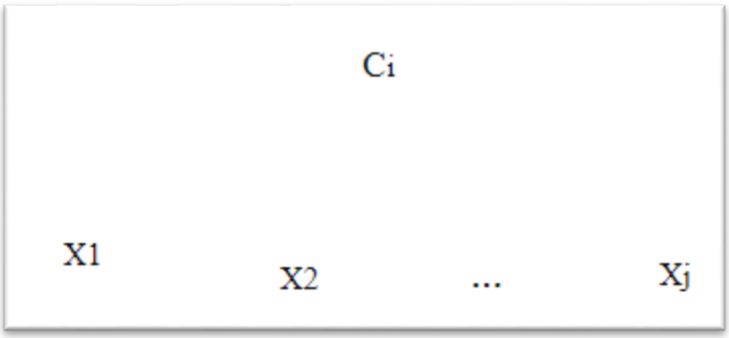

Figure 2.5: Exemple de structure d'un réseau bayésien naïf

Ou :

Ci est le nœud classe et *i* est la ieme classe. *Xj* sont les nœuds des attributs et *j* est le j ieme attribut ou paramètre.

Ce classifier est connu pour ses performances malgré sa simplicité et il dépasse des techniques beaucoup plus sophistiquées même lorsque l'hypothèse d'indépendance est violée. L'hypothèse d'indépendance des variables permet d'écrire la probabilité a posteriori de chaque classe comme l'indique l'équation suivante :

$$P(C_i / x) = P(C_i) \prod_{j=1}^{n} P(x_j / C_i)$$

Par conséquent, en présence d'un ensemble d'apprentissage, la seule opération à faire est de calculer les probabilités conditionnelles en appliquant la règle de décision « *d* » de Bayes comme suit :

d (X) = argmaxclasse P (classe / X)

$$= argmaxclasse\ P\ (Xj / classe)\ P\ (classe)$$

$$= argmaxclasse$$

L'hypothèse d'indépendance entre les attributs utilisés dans le RB naïf est généralement fausse (hypothèse naïve).

Il existe différentes techniques pour assouplir cette hypothèse [17]. Elles consistent à identifier les dépendances conditionnelles entre les attributs. Nous obtenons alors une sous-structure optimale sur les observations en forme d'arbre en adaptant une des méthodes

d'apprentissage de structure, appelée RB augmenté par un arbre (TAN) (fig.2.6).

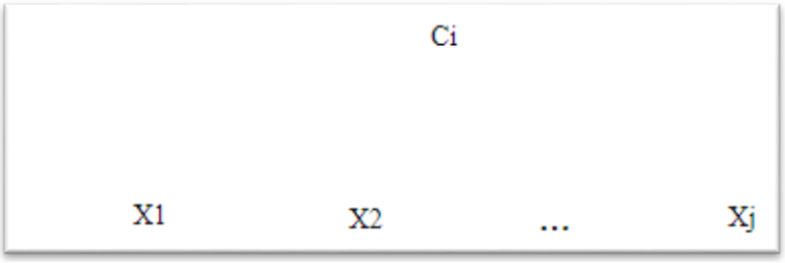

Figure 2.6 : Exemple de structure d'un réseau bayésien naïf augmenté

Où *Ci* est le nœud classe,*i* , la ième classe, *Xj* , les nœuds des attributs et *m*, le nombre d'attributs ou paramètres.

Dont la probabilité à posteriori de chaque classe est l'indiquée par l'équation

Suivante :

$$CR(\ ,\ /\ x\ ) = CP(\ ,\ \prod_{,\ -\ 1}^{\ } xP(\ '_{,}\ /\ Pa\ (\ x\ '_{,}),\ C\ ,\ )$$

Équation

Où)(*i xPa* : est le parent de *xj* par rapports à la classe *i* s'il existe.

Notant que des liens inutiles puissent exister dans un TAN.

Le nombre des liens est fixé à *n-1* en rajoutant les liens existants entre le nœud classe et les attributs. Parfois, il pourrait être sur-adaptés avec les données. Pour remédier à ce problème certaines conditions sont imposées à la structure TAN pour obtenir une sous structure plus optimale appelée réseau bayésien augmenté par un foret (FAN) [18], dont la probabilité à posteriori de chaque classe et celle du TAN indiquée par l'équation.

Cependant une application donnée peut être représentée par plusieurs structures et donc différents paramètres ou probabilités, d'où la nécessité d'apprentissage de structure et de paramètre.

Apprentissage de paramètres et de structure

La construction d'un réseau bayésien consiste à trouver une structure ou un graphe et estimer les paramètres (probabilités conditionnelles).

Cependant devant une très grande base de données, personne ne peut extraire seule la structure adaptée à une telle quantité de données. C'est ici qu'intervient l'apprentissage artificiel.

Apprentissage des paramètres

L'apprentissage de paramètres consiste à supposer que la structure du réseau et fixe et donc à déterminer les probabilités conditionnelles de chaque variable qui se trouve dans le réseau. Les données disposées peuvent être complètes ou incomplètes, discrètes ou continues. Pour chaque cas, l'algorithme d'apprentissage des paramètres diffère. Dans le cas où toutes les variables sont observées et discrètes, la méthode la plus simple et la plus utilisée pour estimer les paramètres est l'estimation statistique de la

probabilité d'un évènement par la fréquence d'apparition de l'évènement dans la base de donnée. Cette méthode est appelée maximum de vraisemblance [40], donnée par l'expression suivante:

$$XP(\quad_i = x_k \;/\; \Pi_i = \pi_{ij}) = \theta_{ijk} = \frac{}{\sum_{k=1}^{ri} \quad ijkN}$$

Le nombre d'occurrences simultanées dans la base de $Xi = xk$ et avec :

$$i \in .....1\ r_i \qquad et \qquad j \in .....1\ q_i$$

Apprentissage de structure

L'apprentissage de structure ayant pour but de trouver le meilleur réseau permettant de représenter les données le mieux possible.

Cependant, la recherche dépend de nombre de variables, d'arcs et de valeurs. Le nombre de structure généré possible à partir de *n* noeuds est très grand, il est donné par la relation suivante :

Soit :)(*nr* le nombre de graphe possible, et *n* le nombre de nœuds existants.

$$nr)(\quad = \sum_{i=1}^{n} (-)_1^{\,i+1}\ C_i^n\ 2^{n(\ -\ i)}(_{ninr}\ -\ )= \quad 20\ (\ )$$

Ainsi pour *n=4* on' a *r (4) =543* et pour *n = 7* on' a *r (7) = 1,4.109.*

Beaucoup de travaux se sont intéressés aux problèmes de l'apprentissage de structures [16]. La aussi comme pour l'apprentissage de paramètres, on a deux cas, selon que les données sont totalement ou partiellement observables.

Pour le premier cas deux familles d'approches ont été proposées. A savoir celles qui sont basées sur des scores pour trouver la configuration optimale d'un réseau bayésien.

Ces algorithmes consistent à parcourir tous les graphes possibles puis associer un score à chaque graphe.

Le graphe qui possède le plus grand score va être sélectionné. Cependant, cette méthode est applicable seulement pour des problèmes à taille limitée (quelques centaines de variables).

Il existe deux types de score : les scores locaux et les scores globaux [19].

La deuxième famille utilise des tests statistiques afin de déterminer les indépendances entre les variables dans le réseau.

Inférences

L'inférence est le calcul de la probabilité de n'importe qu'elle variable d'un modèle probabiliste à partir de l'observation d'une ou de plusieurs autres variables.

Il consiste à propager une ou plusieurs informations au sein de ce réseau, pour en déduire comment sont modifiées les croyances concernant les autres nœuds.

La structure du graphe joue un rôle important dans la complexité de ces calculs ainsi dans le choix de la méthode d'inférence.

On peut distinguer deux catégories d'algorithmes d'inférence [20] : l'inférence exacte et l'inférence approché. Plusieurs méthodes ou algorithmes conçues spécialement pour les problèmes d'inférence exacte pour les réseaux bayésiens

- **Les réseaux de neurones artificiels (RNA)**

❖ Qu'est-ce qu'un réseau de neurones artificiels ?

Un réseau de neurones artificiels, ou Artificial Neural Network en anglais, est un système informatique matériel et / ou logiciel **dont le fonctionnement est calqué sur celui des neurones du cerveau humain**.
Il s'agit là d'une variété de technologie Deep Learning (apprentissage profond), qui fait elle-même partie de la sous-catégorie d'intelligence artificielle du Machine Learning (apprentissage automatique).

❖ Comment fonctionne le réseau de neurones artificiels ?

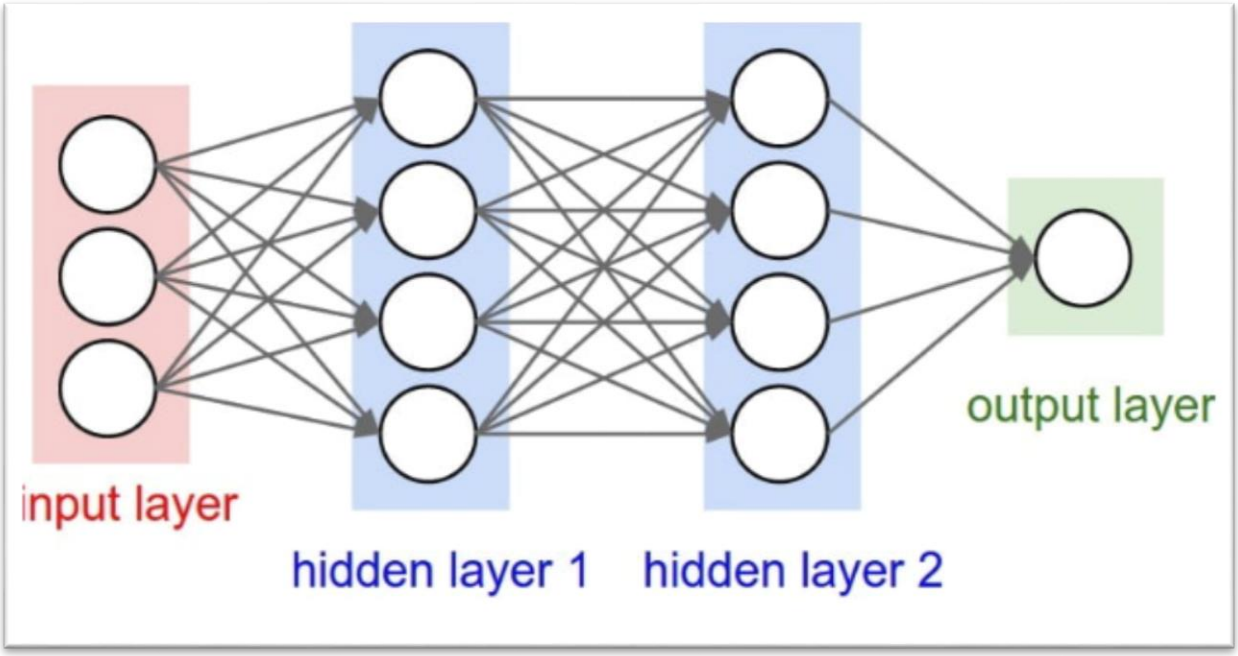

Figure 2.7 : fonctionnement de réseau de neurone

En règle générale, un réseau de neurones repose sur un grand nombre de processeurs opérant en parallèle et organisés en tiers. Le premier tiers reçoit les entrées d'informations brutes, un peu comme les nerfs optiques de l'être humain lorsqu'il traite des signaux visuels.
Par la suite, chaque tiers reçoit les sorties d'informations du tiers précédent**.**
On retrouve le même processus chez l'Homme, lorsque les neurones reçoivent des signaux en provenance des neurones proches du nerf optique. Le dernier tiers, quant à lui, produit les résultats du système.

❖ **Comment le réseau de neurones artificiels apprend ?**

Par le biais d'un algorithme, le réseau de neurones artificiels permet à l'ordinateur d'apprendre à partir de nouvelles données. L'ordinateur doté du réseau de neurones apprend à effectuer une tâche en analysant des exemples pour s'entraîner. Ces exemples ont préalablement été étiquetés afin que le réseau puisse savoir ce dont l s'agit.

Par exemple, un réseau de neurones peut être utilisé pour apprendre à l'ordinateur à reconnaître des objets. Un grand nombre d'objets d'une même catégorie est présenté au réseau de neurones, et l'ordinateur apprendre à reconnaître cet objet sur de nouvelles images en analysant les patterns récurrentes au sein des images d'exemple. Ainsi, en analysant des milliers de photos de chats, le Neural Network apprendra à reconnaître un chat sur n'importe quelle photo.

Contrairement à d'autres types d'algorithmes, les réseaux de neurones ne peuvent pas être programmés directement pour effectuer une tâche. A la manière du cerveau en développement d'un enfant, la seule instruction qu'ils ont est d'apprendre.

On distingue toutefois trois méthodes d'apprentissage distinctes.

Dans le cas de l'apprentissage supervisé**,** l'algorithme s'entraîne sur un ensemble de données étiquetées et se modifie jusqu'à être capable de traiter la basse de donnée pour obtenir le résultat souhaité.

Dans le cas de l'apprentissage non-supervisé**,** les données ne sont pas étiquetées. Le réseau de neurones analyse l'ensemble de données, et une fonction-coût lui indique dans quelle mesure il est éloigné du résultat souhaité.

Le réseau s'adapte alors pour augmenter la précision de l'algorithme.

Enfin, avec la méthode de l'apprentissage renforcé**,** le réseau de neurones est renforcé pour les résultats positifs et sanctionné pour les résultats négatifs. C'est ce qui lui permet d'apprendre au fil du temps, de la même manière qu'un humain apprend progressivement de ses erreurs.

➢ **K-means**

❖ **Définition**

K-means (k-moyennes) est un algorithme non supervisé de clustering**,** populaire en Machine Learning. Lors de cet article, nous allons détailler son fonctionnement et dans quel cas d'usage il peut être appliqué.

❖ **Algorithme particulièrement simple**

Entrée : X (n obs., p variables), K #classes

Initialiser K centres de classes Gk

REPETER

Allocation. Affecter chaque individu à la

classe dont le centre est le plus proche

Représentation. Recalculer les centres de

classes à partir des individus rattachés

JUSQU'À Convergence

Sortie : Une partition des individus

caractérisée par les K centres de classes Gk

K_means avenages et inconvénients

- **Avantages**

Scalabilité : Capacité à traiter les très grandes bases. Seuls les vecteurs des moyennes sont à conserver en mémoire centrale.

Complexité linéaire par rapport au nombre d'observations (pas de calcul des distances deux à deux des individus, cf. CAH).

- **Inconvénients**

Mais lenteur quand même parce que nécessité de faire passer plusieurs fois les observations. L'optimisation aboutit à un minimum local de l'inertie intraclasses W.

La solution dépend du choix initial des centres de classes. La solution peut dépendre de l'ordre des individus (MacQueen)

2.5 Conclusion

Ce chapitre a passé en revue certains des cas d'utilisation de l'apprentissage automatique, des méthodes courantes et des approches populaires utilisées sur le terrain, des langages de programmation appropriés pour l'apprentissage automatique. Parce que l'apprentissage automatique est un domaine en constante innovation, il est important de garder à l'esprit que les algorithmes, les méthodes et les approches continueront de changer

Chapitre3 : Implémentation et Présentation de L'application

3.1. Introduction

On propose un système de classification neuronal pour la détection d'intrusion car les intrusions informatiques présentent un gros risque pour le système on utilise une base de donnée KDDCup99 afin d'établir un système de détection d'intrusions basé sur l'analyse du comportement et permet de les classifier en deux types (anomaly et normal)

Pour cela nous avons montré dans ce chapitre les différentes phases du développement de notre application, en commençant par l'environnement de développement et une description de la base de données KDDCup99, les étapes de prétraitement que nous avons fait sur cette dernière nous avons donnée tous les informations sur MLP (perceptron multicouche), ensuite nous présentons notre modèle de classification suivi par la discussion des résultats obtenus.

3.2. Perceptron multicouche:

Perceptron multicouche

Définition1 : Le perceptron multicouche (PMC) est la deuxième grande famille de réseaux de neurones. Après avoir décrit l'architecture de ces réseaux on va aborder leur apprentissage, et le concept de rétro propagation de l'erreur [21].

Definition2 : Le perceptron multicouche (ou MLP pour *Multi-Layer Perceptron*) est le type de réseau de neurone le plus simple. Celui-ci est un composé de plusieurs unités, appelées « neurones », reliées entre elles par des connexions. A chaque connexion est associée un poids comprit entre 0 et 1. Les neurones sont organisés en couches :

- Une couche d'entrée.
- Une couche de sortie.
- Une ou plusieurs couches cachées [22].

Caractéristique de perceptron multicouche

Le Perceptron multicouche se caractérise par:

- Il est basé sur le modèle du Perceptron.
- Il a plusieurs couches de neurones liées entre elles.
- Chaque couche a un ou plusieurs neurones

Le Perceptron simple ne pouvait classifier que des données séparées par un hyperplan. Nous allons passer à des données plus complexes séparables par des hyper-surfaces.

Nous allons donc travailler avec une couche d'entrée, une couche de sortie et une ou plusieurs couches cachées. En pratique un réseau 3 couches suffit presque toujours. [23]

3.2.1 Fonctionnement

Les mathématiques du Perceptron multicouche sont similaires à celles du Perceptron. La phase d'entrainement également.

On présente donc un essai à la couche d'entrée et on calcule les sorties de chaque neurone jusqu'à la couche de sortie en utilisant la formule bien connue maintenant:

$$S = b + w_1.x_1 + w_2.x_2 + \cdots + w_n.x_n = b + \sum_{i=1}^{n} w_i.x_i$$

Oui mais comment passe t'on d'une couche à l'autre?

La géométrie du réseau ressemble à celle-ci dans laquelle chaque cercle représente un neurone formel complet (avec la fonction de transfert):

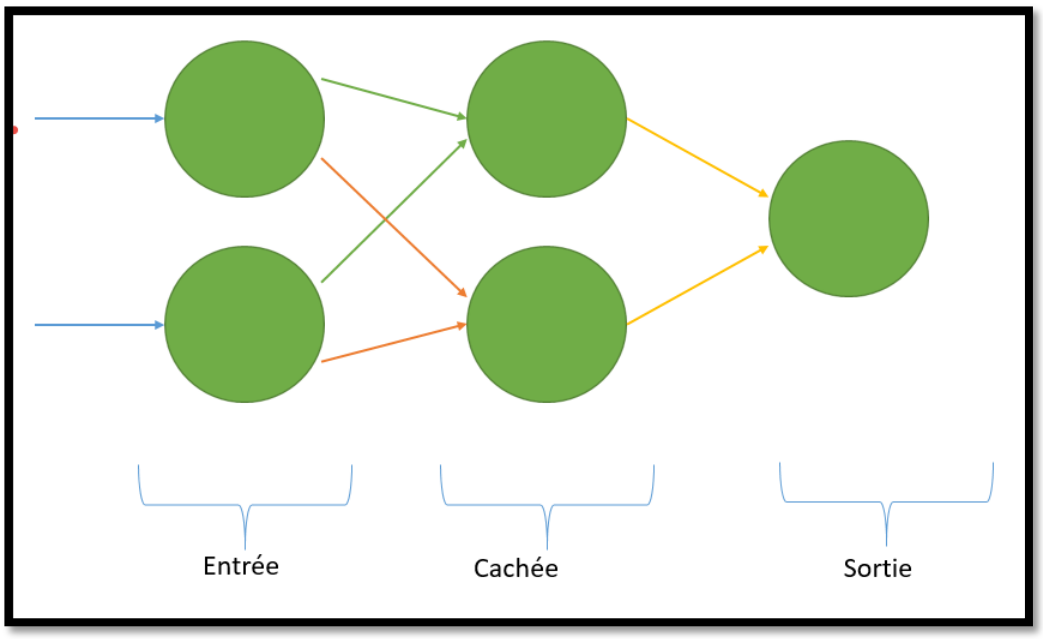

Figure3.1 : perceptron multicouche

Comme on le voit sur le schéma, chaque sortie de la couche N constitue une entrée de la couche N+1. Chaque neurone de la couche cachée par exemple, a deux entrées.

Nous utilisons la classe **DeepNetwork** qui modélise notre réseau. Cette classe abrite pour l'essentiel une collection de **Couche** qui modélise une couche particulière. On remarque que chaque couche peut avoir sa propre fonction de transfert.

Couche abrite à son tour une collection de **Neurone** qui modélise un neurone.

L'entrainement du réseau est réalisé dans **DeepNetwork.Entrainer**. La structure du code est similaire à celle examinée dans le cas du Perceptron simple.

La méthode **Calculer()** calcule un état du réseau, c'est à dire les sorties des neurones de la couche de sortie étant donnée un état de la couche d'entrée.

Comme dans le cas du Perceptron simple on obtient une sortie que l'on compare avec les valeurs ciblées. On calcule de même un *ecart*.

Le problème est alors de savoir redistribuer l'erreur obtenue sur les neurones et les biais de la couche de sortie, mais également des couches cachées.

L'algorithme standard s'appelle **rétro-propagation** (back-propagation) découvert dans les années 70. Il est appelé ainsi car il est composé de deux mouvements:

1. Calcul des états du réseau en partant de la couche d'entrée
2. Correction des états étant donnée l'erreur du réseau en partant de la couche de sortie

   Nous avons vu la première phase. Voyons la suivante.

   Les mathématiques de la phase d'apprentissage de ce réseau sont plus complexe que ce qui a été vu dans le cas du Perceptron simple. On a cette fois besoin de la dérivée de la fonction de transfert qui doit donc être dérivable.

   On peut trouver les formules avec un exemple de calcul ici:

- https://www.youtube.com/watch?v=I2I5ztVfUSE
- https://www4.rgu.ac.uk/files/chapter3%20-%20bp.pdf
- http://galaxy.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html

  Et un exemple de démonstration ici:

  http://alp.developpez.com/tutoriels/intelligence-artificielle/reseaux-de-neurones/

  Si vous décidez d'écrire votre propre code (et je vous le conseille), ces liens vont beaucoup vous aider à comprendre les calculs [24].

3.2.2 L'architecture de perceptron multicouche

- **Le perceptron multicouche**

Désigné par le sigle MLP (pour *Multi-layer Perceptron*), le perceptron multicouche se compose d'une couche d'entrée, d'une couche de sortie et d'une ou plusieurs couches cachées. Si le réseau possède n couches, alors il possède n-1 matrice de poids (une entre chaque suite de couches). Le MLP doté d'une couche cachée est théoriquement un approximateur universel de fonctions. En théorie, il suffit donc d'ajouter un nombre de neurones suffisant au niveau de la couche cachée pour approximer n'importe quelle fonction non linéaire [25].

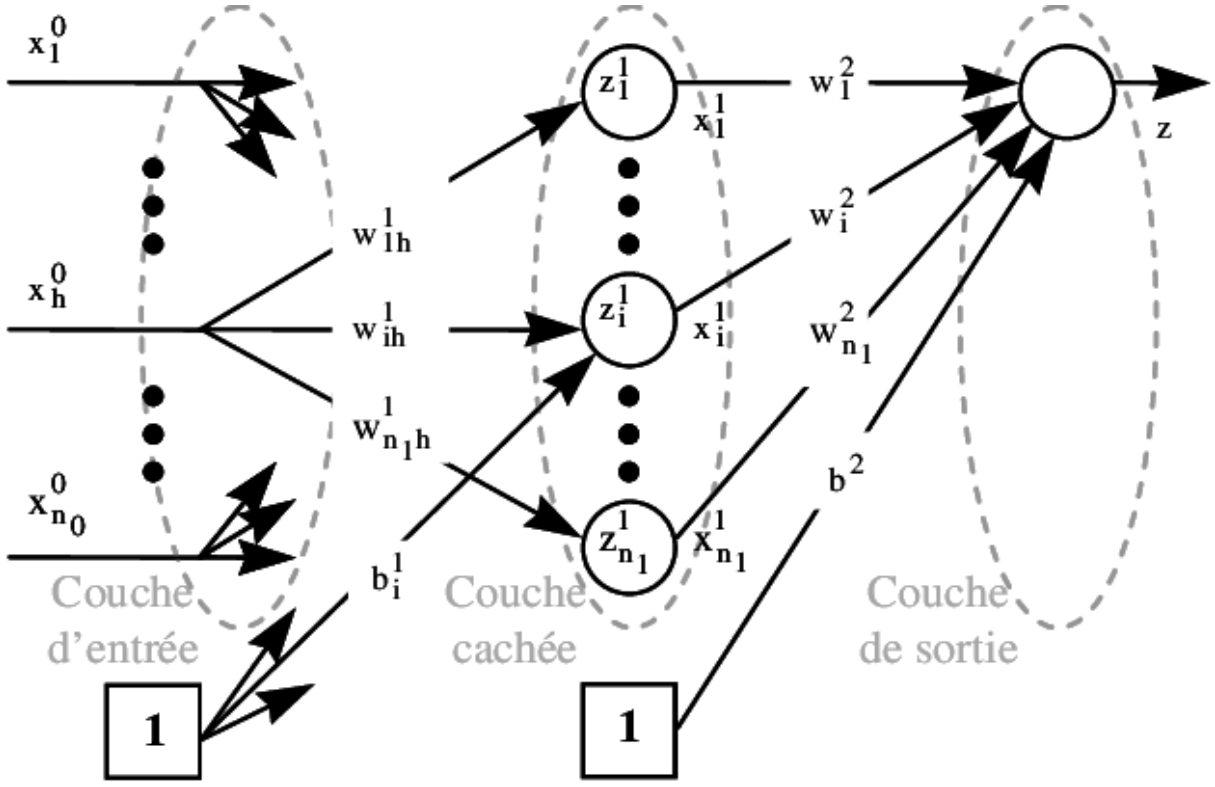

Figure 3.2 : l'architecture de MLP

Perceptron multicouche – Avantages et inconvénients Avantages :

- Classifieur très précis (si bien paramétré).
- Incrémentalité.
- Scalabilité (capacité à être mis en œuvre sur de grandes bases).

Inconvénients :

- Modèle boîte noire (causalité descripteur – variable à prédire).

- Difficulté de paramétrage (nombre de neurones dans la couche cachée).
- Problème de convergence (optimum local).

Danger de sur-apprentissage (trop de neurones dans la couche cachée) [26].

3.3 Description de la base KDDcup99 et prétraitement :

Depuis 1999 l'ensemble de données KDD'99 (KDD'99, 1999) est devenu la base de données la plus utilisée pour l'évaluation des systèmes de détection d'intrusion. Cet ensemble de données a été préparé par Stolfo et al. (2000), où ils ont utilisé des données extraites de l'ensemble de données d'évaluation des systèmes de détection d'intrusion le DARPA'98 (Lippmann et al, 2000).

Le DARPA'98 contient environ 4 gigabits d'enregistrements binaires compressés des données tcpdump collectées durant 7 semaines de surveillance de trafic réseau qui ont été transformées en environ 5 millions d'enregistrements de connexion, où chaque connexion est composée d'environ 100 octets. Les deux semaines des données de test représentent environ 2 millions d'enregistrements de connexion. L'ensemble de données d'apprentissage « KDD training data set » se compose d'environ 4,9 millions d'enregistrements de connexion dont chacun contient 41 caractéristiques ainsi qu'une étiquette qui indique le type de connexion normal ou une attaque (le type spécifique d'attaque). Les attaques simulées appartiennent à l'une des quatre catégories suivantes:

- **L'attaque de déni de service « Denial of Service Attack (DoS) » :** est une attaque dans laquelle l'attaquant essaye de rendre la mémoire d'un système ou la bande passante d'un réseau trop occupée ou trop chargée pour gérer les demandes légitimes, afin d'empêcher les utilisateurs légitimes d'accès à une machine.
- **L'attaque de passage d'un utilisateur à un super utilisateur « User to Root Attack (U2R) » :** est une attaque dans laquelle l'attaquant commence par un accès à un compte utilisateur normal sur le système (peut-être acquise par des mots de passe capturés, une attaque par dictionnaire… etc.) dans le but d'exploiter certaines vulnérabilités pour obtenir un accès Root sur le système.
- **L'attaque distance à local « Remote to Local Attack (R2L) » :** se produit quand un attaquant qui a la capacité d'envoyer des paquets vers une machine sur un réseau, mais qui n'a pas de compte sur cette machine. Il exploite certaines vulnérabilités afin d'obtenir

un accès local en tant qu'utilisateur de cette machine.

- **L'attaque d'exploration « Probing Attack » :** elle vise de rassembler des informations sur un réseau d'ordinateurs dans le but de contourner les contrôles de sécurité [27].

#	Nom de la propriété	Type de la propriété	Description de la propriété
1	Duration	Continu	Longueur de la connexion (second)
2	Protocol_type	Discret	Type de protocole, e.g. tcp, udp, etc.
3	Service	Discret	Service réseau de destination, e.g., http, telnet, etc.
4	Flag	Discret	Statut normal ou erreur de la connexion
5	Src_bytes	Continu	Nombre d'octets de données de la source à la destination
6	Dst_bytes	Continu	Nombre d'octets de données de la destination à la source
7	Land	Discret	1 si une connexion est de / vers le même hôte / port; 0 sinon
8	Wrong_fragment	Continu	Nombre de fragments « erronées »

9	Urgent	Discret	Nombre de paquets urgents
10	Hot	Discret	Nombre d'indicateurs "hot"
11	Num_failed_logins	Discret	Nombre de tentatives de connexion échouées
12	Logged_in	Discret	1 si un succès de se connecter, 0 sinon
13	Num_compromised	Discret	Nombre de conditions compromises
14	Root_shell	Discret	1 si le root Shell est obtenu; 0 autrement
15	Su_attempted	Discret	1 si la commande " su root " a été tentée, sinon 0
16	Num_root	Discret	Nombre de " root " ont accédé
17	Num_file_creations	Discret	Nombre d'opérations de création de fichiers
18	Num_shells	Discret	Nombre d'invités du shell
19	Num_access_files	Discret	Nombre d'opérations sur les fichiers de contrôle d'accès
20	Num_outbound_cmds	Discret	Nombre de commandes sortantes dans une session FTP
21	Is_host_login	Discret	1 si la connexion appartient à la liste du 'hot' ; 0 sinon
22	Is_guest_login	Discret	1 si le login est un login "guest", sinon 0
23	Count	Discret	nombre de connexions vers la même machine que la connexion en cours dans les deux dernières secondes
24	Srv_count	Discret	Nombre de connexions pour le même service que la connexion en cours dans les deux dernières secondes
25	Serror_rate	Discret	% Des connexions qui ont des erreurs "SYN" (connexions de la même machine)
26	Srv_serror_rate	Discret	% Des connexions qui ont des erreurs "SYN" (connexions du même service)

27	Rerror_rate	Discret	% Des connexions qui ont des erreurs "REJ" (connexions de la même machine)
28	Srv_rerror_rate	Discret	% Des connexions qui ont des erreurs "REJ" (connexions du même service)
29	Same_srv_rate	Discret	% des connexions aux mêmes services
30	Diff_srv_rate	Discret	% de connexions aux différents services
31	Srv_diff_host_rate	Discret	% de connexions aux différentes machines
32	Dst_host_count	Discret	compteur pour la machine de destination
33	Dst_host_srv_count	Discret	Srv_count pour la destination host
34	Dst_host_same_srv_rate	Discret	Same_srv_rate pour la destination host
35	Dst_host_diff_srv_rate	Discret	Diff_srv_rate pour la destination host
36	Dst_host_same_src_port_rate	Discret	Same_src_port_rate pour la destination host
37	Dst host srv diff host rate	Discret	Diff_host_rate pour la destination host
38	Dst_host_serror_rate	Discret	Serror_rate pour la destination host
39	Dst_host_srv_serror_rate	Discret	Srv_serror_rate pour la destination host
40	Dst_host_rerror_rate	Discret	Rerror_rate pour la destination host
41	Dst_host_srv_rerror_rate	Discret	Srv_serror_rate pour la destination host

Tableau1 : les propriétés de KDDCup99

Comme le DARPA les enregistrements du KDD'99 sont correctement étiquetés soit comme un type spécifique d'attaque ou comme normal. Normal représenté toute connexion non intrusive (les comportements habituels d'un utilisateur) telle que la consultation d'une page web ou le téléchargement des applications et des fichiers...etc. Toutes les attaques du KDD'99 appartiennent à l'une des quatre classes d'attaques précédemment présentées dans la description du DARPA (DoS, Probing, R2L, U2R). Le KDD'99 contient 39 attaques où 17 attaques n'existent que dans les données de test. Le tableau suivant résume la classification des différentes attaques du KDD'99 en quatre catégories d'attaques.

Les catégories des attaques	Type d'attaque		
	Attaque existe dans les données d'apprentissage et de test	**Attaque existe que dans les données d'apprentissage**	**Attaque existe que dans les données de test**
DOS	back, land, neptune, pod, smurf, teardrop		**apache2, mailbomb, processtable, udpstorm**
Probe	ipsweep, nmap, portsweep, satan		**mscan, saint**
R2L	ftp_write, guess_passwd, imap, multihop, phf, warezmaster	spy, warezclient	**named, sendmail, snmpgetattack, snmpguess, worm, xlock, xsnoop**
U2R	**buffer_overflow, loadmodule, perl, rootkit**		**httptunnel, ps, sqlattack, xterm**

Tableau2 : La classification des attaques du KDDCup 99

Prétraitement de l'ensemble de données de la base NSL-KDD : Les Données de KDDCup99 contient 41 propriétés qui peuvent être classées en trois groupes: **Les caractéristiques de base, les caractéristiques du trafic, caractéristiques du contenu**.

- ✓ **La sélection d'attribut :**

Vu que la taille de la base de données KDDCup99 est très importante, dans notre cas (41 attributs et 22544 enregistrements pour l'entrainement),donc le travail sur tous les instances pour générer le modèle de classification sera très fastidieux et peut affecter considérablement les performances de l'algorithme d'apprentissage en matières de temps d'exécution et consommation des ressources systèmes, en plus les instances de la base de données ne seront pas tous utilisés dans la classification effectuée par l'IDS pour détecter les attaques, certains étant plus pertinents que d'autres.

Pour ces raisons, et dans le cadre de notre projet, la sélection des attributs constitue une tache non obligatoire mais très importante pour extraire des sous-ensembles des attributs en préservant les plus significatives et pertinents dont le but est de faire une classification supervisée de l'ensemble de données KDDCup99 en deux catégories (Normal et anomaly) et de réduire le coût et le temps nécessaire pour l'opération d'apprentissage.

On prend comme un échantillon pour faire un système de détection d'anomalie 1127 lignes et 41 attributs pour notre projet fin d'étude on applique une méthode de sélection des caractéristiques nommé **ClassifierSubsetEval** pour réduire le nombre des attributs cette méthode est existe dans **Weka** efficace, rapide, et donne des résultats utilisables avant de faire cette étape nous avons réduire le nombre des lignes on utilise la méthode **Resample** (KDDCup99 contient plus de 20000 nous avons réduire jusqu'à 1127 instances) le figure suivant donne une idée sur la phase de sélection d'attributs 1

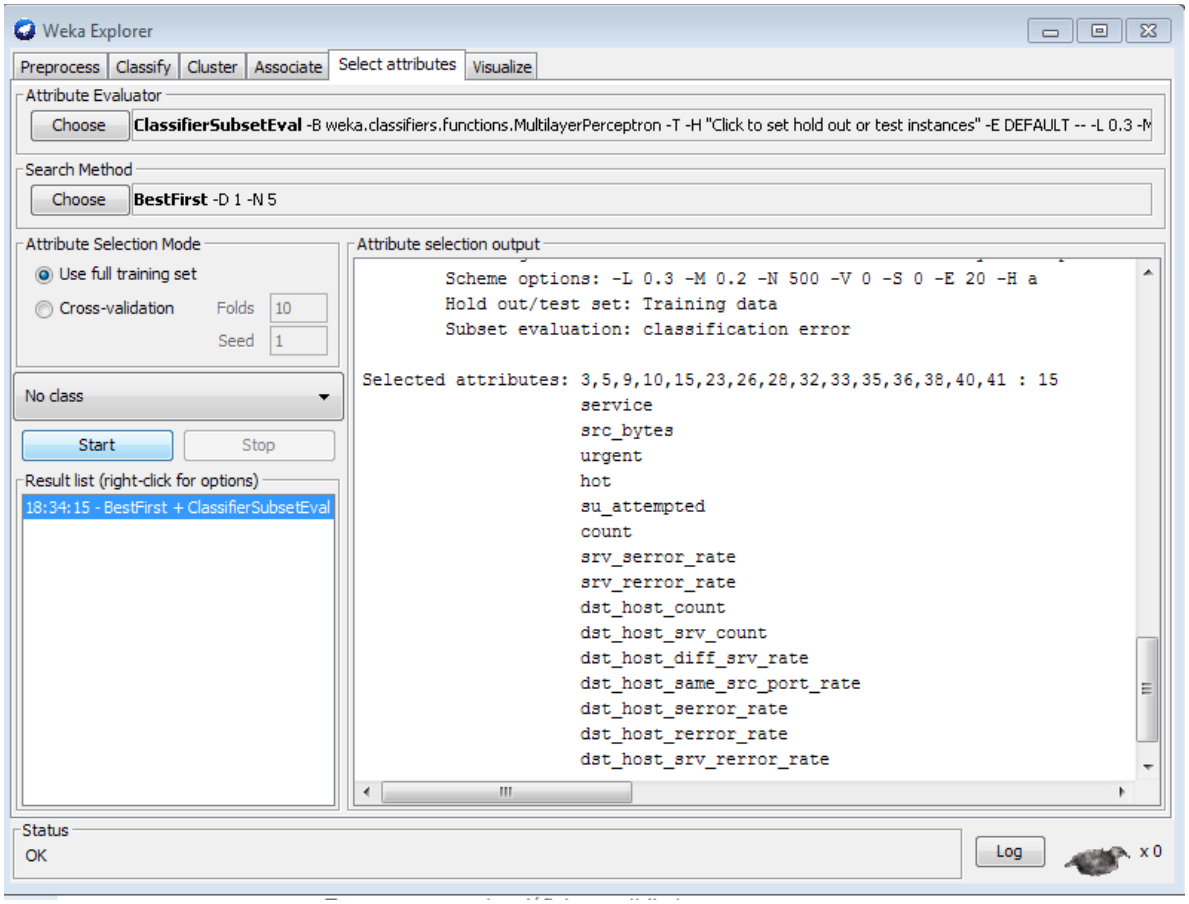

Figure 3.3 : la phase de sélection des caractéristiques

On utilise **Weka** pour la phase de sélection des caractéristiques on appliques les étapes suivant :

Select attribu → Attribute Evaluato → chose

↓

ClassifierSubsetEval(Multi Layer Perceptron)

↓

15attributs (resultat)

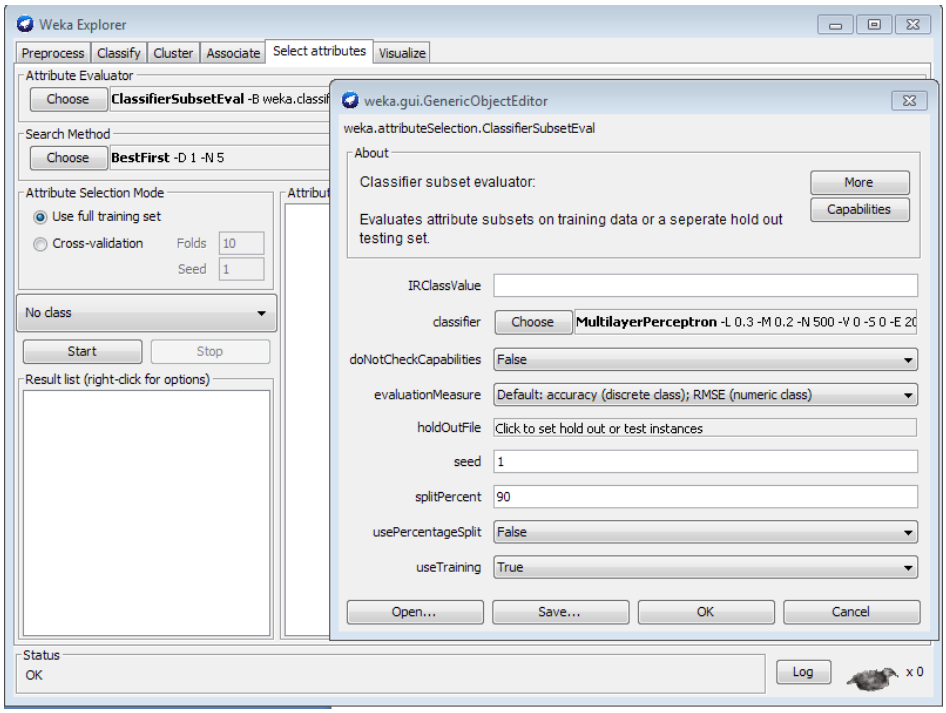

Figure3.4 : ClassifierSubsetEval (Multi layer perceptron)

Apres l'étape de sélection nous avons obtenir 15 attributs et 1127 instances comme une résultat qui nous avons faire une discussion et Expérimentation et résultats.

3.4 : Expérimentation et résultat :

3.4.1 : Environnement de programmation :

Nous avons choisi l'environnement de programmation Weka3.9.3 pour Windows 7 afin d'implémenté les sélections des caractéristiques et l'apprentissage et de test de notre modèle de détection d'intrusions.

Weka Machine Learning Project (GNU Public License).

•Format ARFF.

•Beaucoup de méthodes de classification supervisées.

•Quelques méthodes de classification automatiques.

•Quelques méthodes de recherches de sous-ensembles fréquents.

•Fichiers de résultats standardisés .

Weka est maintenant installé sur votre compte. Après l'avoir lancé, vous obtenez la fenêtre intitulée Weka GUI Chooser : choisissez l'Explorer. La nouvelle fenêtre qui s'ouvre alors (Weka Knowledge Explorer) présente six onglets :

Preprocess : pour choisir un fichier, inspecter et préparer les données.

Classify : pour choisir, appliquer et tester différents algorithmes de classification : là, il s'agit d'algorithmes de classification supervisée.

Cluster : pour choisir, appliquer et tester les algorithmes de segmentation.

Associate : pour appliquer l'algorithme de génération de règles d'association.

Select Attributes : pour choisir les attributs les plus prometteurs.

Visualize : pour afficher (en deux dimensions) certains attributs en fonctions d'autres.

 Notepad++ est un éditeur de texte générique codé en C++, qui intègre la coloration syntaxique de code source pour les langages et fichiers [28].

Pour l'implémentation nous avons utilisé un micro-ordinateur ayant les caractéristiques techniques représentées dans le tableau suivant:

3.4.2 Description de l'application :

- **Fenêtre principale :** qui compose plusieurs boutons qui s'indique des autres fenêtres.

Figure 3.5. weka3.9.4

- ✓ **Fenêtre Explorer :** qui contient Preprocess, classify, cluster, associate, select attributes, visualize**.**

Fenêtre Preprocess : On fait l'étape de réduction des instances dans notre projet comme une première phase le figure suivant explique (figure3.6) :

On utilise Resample comme une méthode pour réduire les lignes avec samplesizeparcent=5

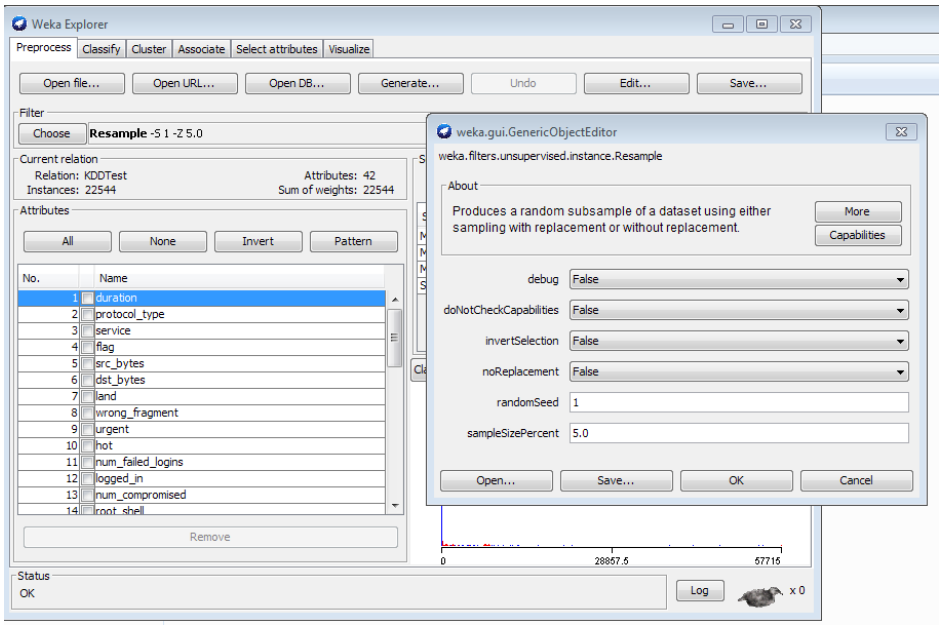

Figure3.6 : réduire les instances de KDDCup99

Fenêtre Sélection d'attributs: l'opération de sélection d'attributs est optionnelle, si cette option est activé nous allons se dirigé vers une autre fenêtre qui nous permet nous :

- ✓ Utilisé la méthode de sélection des caractéristiques ClassifierSubsetEval dans notre projet les résultats est 15 attribut (3,5,9,10,15,23,26,28,32,33,35,36,38,40,41).

Comme on a déjà mentionné ci-dessus, on va s'intéresser à la classification. Dans weka, on peut trouver tous les algorithmes correspondant dans l'onglet classify. Après avoir chargé les données dans l'outil, on applique les différentes fonctions destinées pour cette étude. Weka offre quatre options pour faire l'évaluation de la performance du modèle appris, dans cette expérience, on ne va se pencher que sur deux d'entre elles. Dans un premier temps, appliquer les algorithmes sur la totalité des données (use training set) en conservant les paramètres par défaut. Ensuite, on va employer la validation croisée (cross validation), l'ensemble d'apprentissage va être subdivisé en 10 parties (fold =10) ; l'algorithme va apprendre 10 fois sur 9 parties et la dernière partie sert à évaluer le modèle, puis les 10 évaluations sont combinés. La figure ci-après illustre l'onglet et les options de tests.

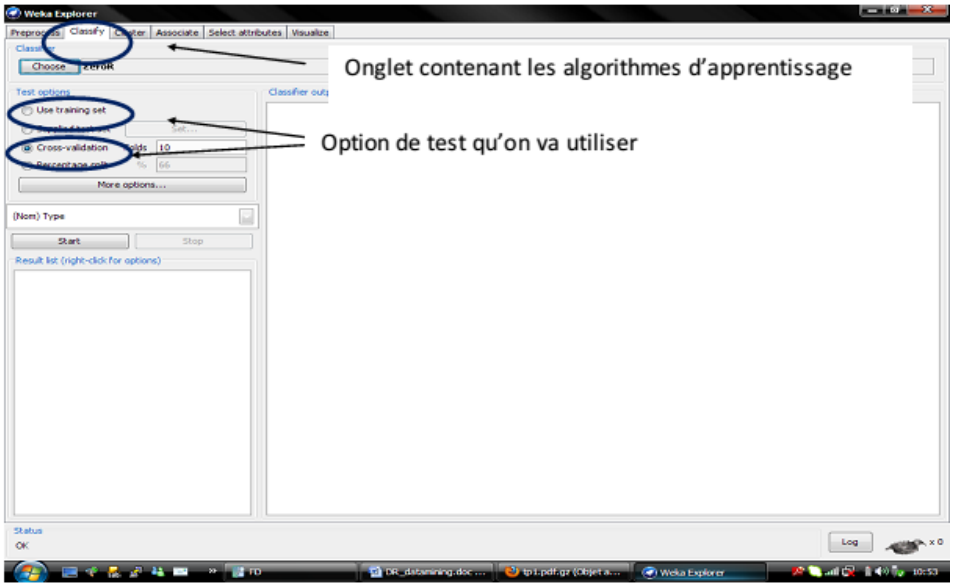

Figure 3.7 : classification

Fenêtre classify : pour faire la classification on choisi deux modèles nous avons utilise MLP et j48 (arbre de décision).

Méthode de classification:

- ✓ Perceptron multicouche (Multi Layer Perceptron) : est la fonction implémentant l'algorithme de perceptron multicouche dans weka. On utilisera les paramètres par défaut à savoir le nombre des couches cachées égale à a=(nombre d'attributs+nombre de classes)/2, le temps de traitement à 500

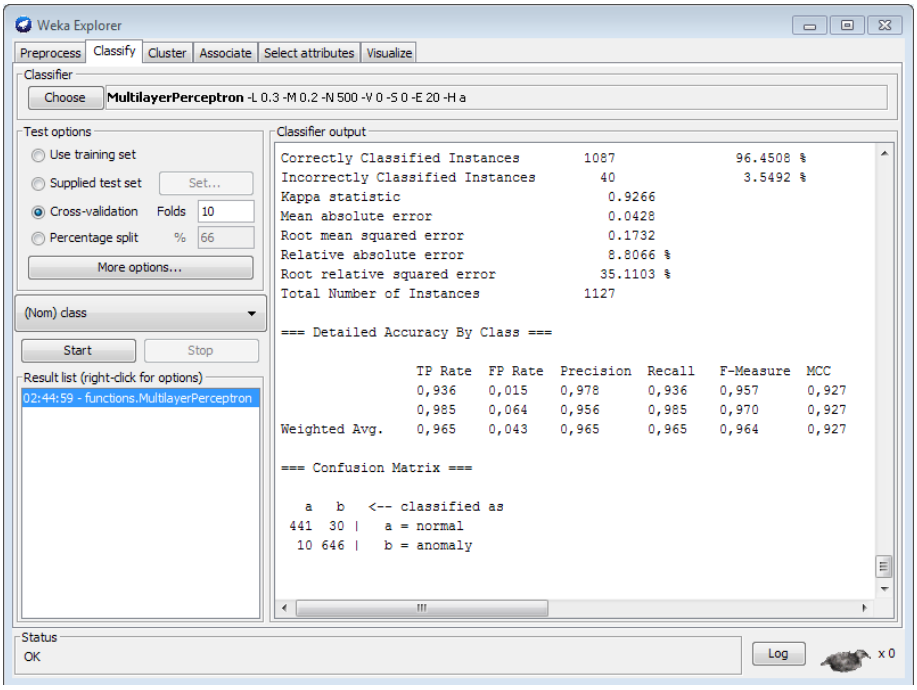

Figure3.8 : taux de classification MLP

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances 1087 96.4508 %

Incorrectly Classified Instances 40 3.5492 %

Kappa statistic 0.9266

Mean absolute error 0.0428

Root mean squared error 0.1732

Relative absolute error 8.8066 %

Root relative squared error 35.1103 %

Total Number of Instances 1127

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0,936	0,015	0,978	0,936	0,957	0,927	0,987	0,987	normal
0,985	0,064	0,956	0,985	0,970	0,927	0,987	0,986	anomaly
0,965	0,043	0,965	0,965	0,964	0,927	0,987	0,986	

F-Measure = 2 × Recall × Precision/ Recall + Precision

=== Confusion Matrix ===

a b <-- classified as

441 30 | a = normal

10 646 | b = anomaly

Donc le taux de classification pour perceptron multicouche est **0.964**

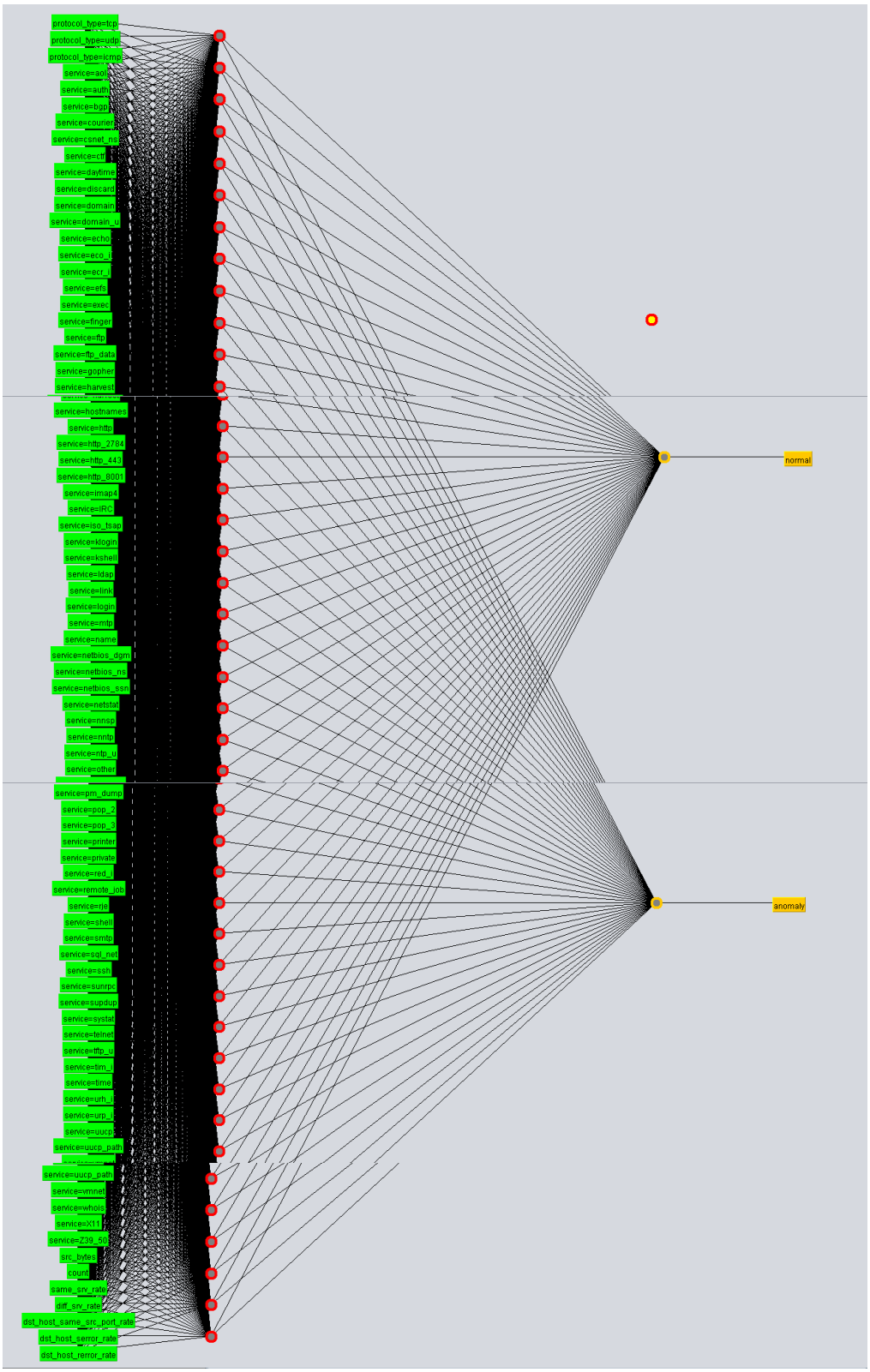

Figure3.9: réseau de neurone MLP

- ✓ Arbre de décision (Tree J48) : L'arbre de décision est une représentation graphique d'une procédure de classification. Dans weka, cet algorithme est représenté par la fonction J48.

Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances 1076 95.4747 %

Incorrectly Classified Instances 51 4.5253 %

Kappa statistic 0.9063

Mean absolute error 0.064

Root mean squared error 0.2003

Relative absolute error 13.1462 %

Root relative squared error 40.6128 %

Total Number of Instances 1127

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0,919	0,020	0,971	0,919	0,952	0,918	0,944	0,907	normal
0,980	0,081	0,944	0,974	0,980	0,918	0,962	0,907	anomaly
0,955	0,055	0,955	0,960	0,95	0,918	0,955	0,907	

=== Confusion Matrix ===

a b <-- classified as

443 28 | a = normal

17 639 | b = anomaly

Le taux de classification de l'arbre de décision j48 est **0.95**

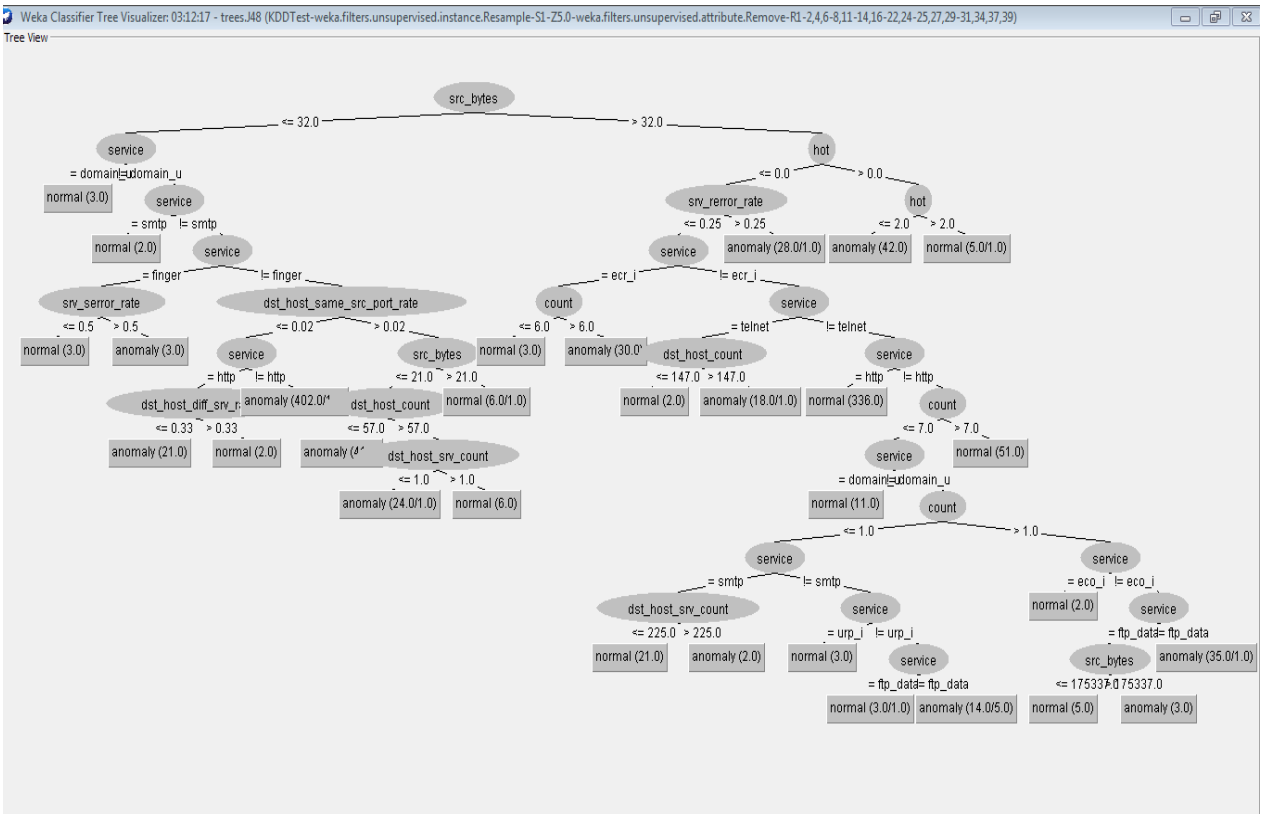

Figure 3.10: L'arbre de décision j48

Méthode dévaluation :

Après la phase de classification on doit parler sur l'étape d'evaluation on utilise deux algorithme MLP et j48 on va faire une petit comparaison sur les resultats donnée on utilise ausssi une base donnée KDDCup99 qui contient 1127 instances et 15 attributs en plus la class (normal et anomaly) on va faire tous dans Weka (Experimenter) le nombre de répétition de cette opération égale 10 et les figures suivants expliquer bien l'étape d'évaluation avec des résultats détailler :

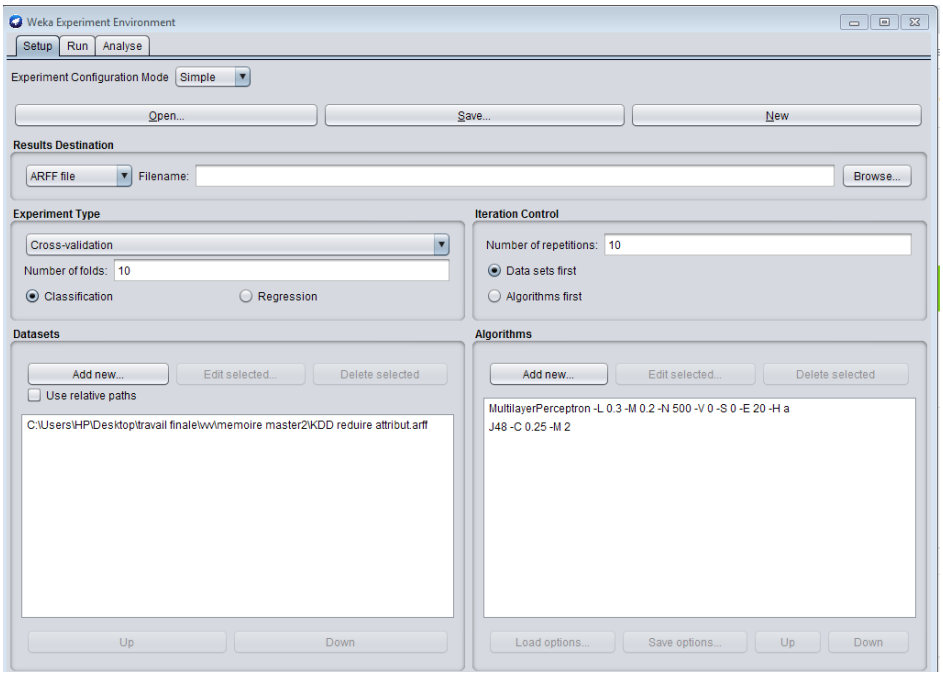

Figure 3.11 : méthode de validation

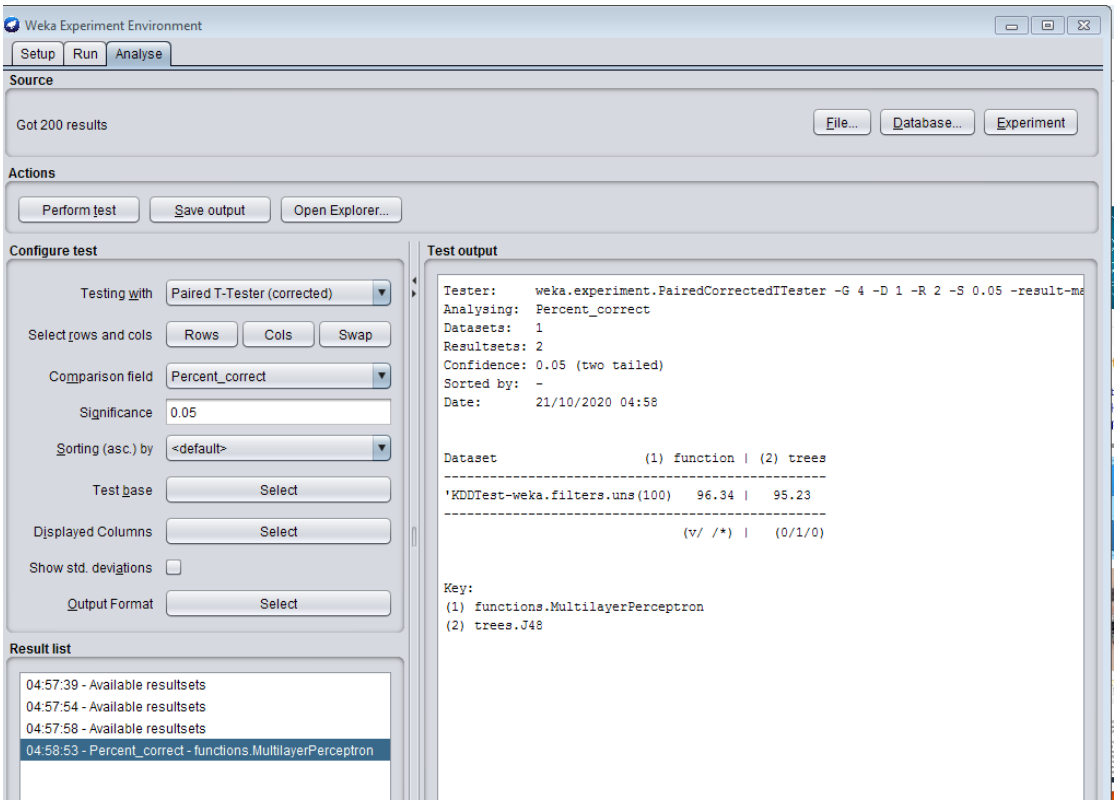

Figure 3.12 : l phase d'analyse

Nous avons remarqué que le taux de validation de j48 < MLP et les résultats expliquer clairement que l'importance de chaque algorithme

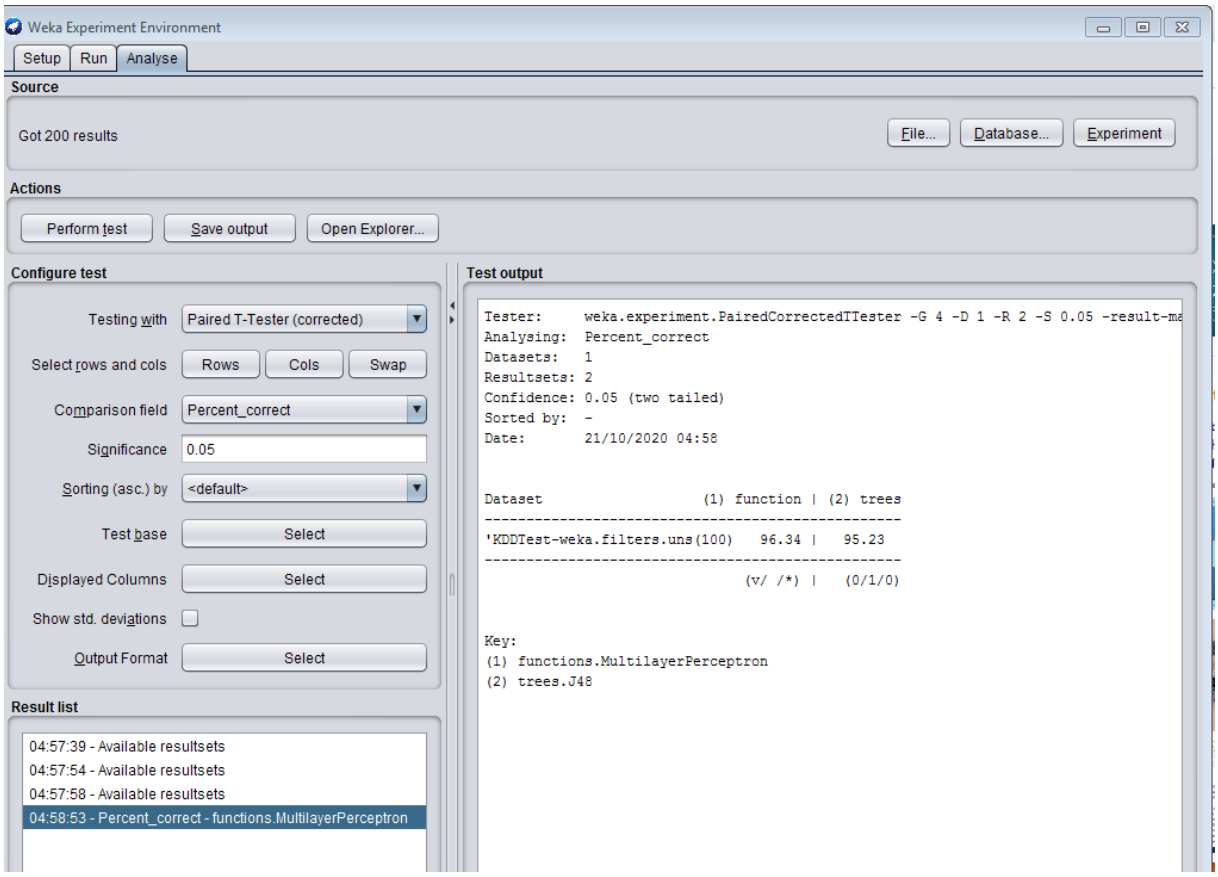

Figure 3.13 : La phase de validation entre j48 et MLP

Fmeasure c'est le taux de classification, donc plus il est élevé, plus c'est bien dans notre projet nous avons remarqué que le taux de MLP et plus élevé que j48 donc le réseau de neurone est meilleur que j48

3.5 Conclusion

Nous avons abordé dans ce dernier chapitre la base de données KDDCup99 qui a été utilisée pour l'apprentissage et test du modèle de détection d'intrusions généré basé sur les réseaux de neurones artificiels. Nous avons aussi expliqué le fonctionnement de l'application implémentée pour mettre en œuvre ce modèle. Ensuite, nous présentons les résultats obtenus après plusieurs valeurs de paramètres du MLP, nous avons parle aussi sur j48 et faire des comparaisons Les résultats ont prouvé l'efficacité des modèles à base de réseaux de neurones dans le domaine de la détection d'intrusions, Les résultats ont également montré l'importance des réseaux de neurones.

Conclusion générale et perspectives

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

Nous avons aussi utilisé la base de données KDDCup99 comme source de données, et nous avons fait un modèle de classification binaire (deux classes : normal et anomaly) pour classifier les données en deux classes : attaque ou normal, à l'aide de perceptron multicouches(MLP) qui est le modèle de réseaux de neurones le plus approprié pour la classification des données non linéairement séparables.

Nous avons faire la sélection des caractéristiques pour réduire le nombre des attributs fait aussi la classification, validation, test…ect.

L'importance de faire un système détection d'anomalie pour minimiser le nombre des risques et protéger nos données.

Références

[1]http://pmb-int.cuniv-aintemouchent.dz/memoire/mathématique/2018/4323-4324/memoire.pdf

[2] Les systèmes de détection d intrusions.pdf (Thierry Evangelista Dunod – ISBN 2 10 007257 9)

[3] H. Debar. Wespi, a revised taxonomy for intrusion detection systems,. 1999.

[4] Ahmim Ahmed. Systéme de détection d'intrusion adaptatif et distribué

[5]https://www.researchgate.net/profile/Ludovic_Me/publication/228878920_Les_systemes_de_detection_d'intrusions_principes_algorithmiques/links/09e41514740ec44fe2000000/Les-systemes-de-detection-dintrusions-principes-algorithmiques.pdf

[6] Metomo JOSEPH BERTRAND RAPHAËL Publié le 10/10/2017 à 13:17:20

[7] Metomo JOSEPH BERTRAND RAPHAËL Publié le 10/10/2017 à 13:17:20

[8] Metomo JOSEPH BERTRAND RAPHAËL Publié le 10/10/2017 à 13:17:20

[9].https://docplayer.fr/60083260-21-10-2011-introduction-apprentissage-rappel-1-svm principe-de-fonctionnement-general.html.

[10].https://docplayer.fr/60083260-21-10-2011-introduction-apprentissage-rappel-1-svm-principe-de-fonctionnement-general.html.

[11],[12],[13].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz.

[14].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[15].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[16].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[17].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[18].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[19].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[20].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[21].https://webcache.googleusercontent.com/search?q=cache:M1mD6uo8LIcJ:https://dl.ummto.dz/bitstream/handle/ummto/593/Zaabot%2520Zohra.pdf%3Fsequence%3D1%26isAllowed%3Dy+&cd=4&hl=ar&ct=clnk&gl=dz

[22]. https://www.lebigdata.fr/reseau-de-neurones-artificiels-definition

[23]. http://eric.univ-lyon2.fr/~ricco/cours/slides/classif_centres_mobiles.pdf

[24] https://www.becoz.org/these/memoirehtml/ch06s04.html.
[25] https://bloommagazine.home.blog/2019/02/06/fonctionnement-du-perceptron-multicouche/.
[26] https://amethyste16.wordpress.com/2015/10/23/reseau-de-neurones-le-perceptron-multicouche/
[27] https://amethyste16.wordpress.com/2015/10/23/reseau-de-neurones-le-perceptron-multicouche/
[28] http://www.scilogs.fr/intelligence-mecanique/architecture-des-reseaux-de-neurones-reseaux-de-neurones-artificiels-classiques-2-3/

Bibliographies

- [And80] : J. Anderson. Computer security threat monitoring and surveillance, 1980.
- [Den87] :Dorothy E. Denning. An intrusion-detection model. *IEEE Transactions on software engeneering*, SE-13 :222–232, 1987.
- [Sma88] :Stephen E. Smaha. Haystack : An Intrusion Detection System. In *Fourth Aerospace Computer Security Applications Conference*, pages 37–44, Tracor Applied Science Inc., Austin, TX, 1988.
- [BK88]: David S. Bauer and Michael E. Koblentz. NIDX – an expert system for real-time network intrusion detection. In *Proceedings of the ComputerNetworking Symposium*, pages 98–106, Washington, DC, 1988. IEEE Computer Society Press.
- [SSHW88]: M. M. Sebring, E. Shellhouse, M. E. Hanna, and R. A. Whitehurst. Expert system in intrusion detection : A case study. In *Proceedings of the 11th National Computer Security Conference*, pages 74–81, 1988.
- [LSM] :Wenke Lee, Salvatore J. Stolfo, and Kui W. Mok. Algorithms for mining system audit data.
- [LSM99] :Wenke Lee, Salvatore J. Stolfo, and Kui W. Mok. A data mining framework for building intrusion detection models. In IEEE Symposium on Security and Privacy, pages 120–132, 1999.
- [LNY+00] :Wenke Lee, Rahul A. Nimbalkar, Kam K. Yee, Sunil B. Patil, Pragneshkumar H. Desai, Thuan T. Tran, and Salvatore J. Stolfo. A data mining and CIDF based approach for detecting novel and distributed intrusions. In Recent Advances in Intrusion Detection, pages 49–65, 2000.
- [JMKM99] :W. Jansen, P. Mell, T. Karygiannis, and D. Marks. Applying mobile agents to intrusion detection and response, 1999.
- [BFFI+98] :Jai Sundar Balasubramaniyan, Jose Omar Farcia-Fernandez, David Isacoff, Eugene Spafford, and Diego Zamboni. An architecture for intrusion detection using autonomous agents. Technical report, November 1998.
- [HWHM98] :G. Helmer, J. Wong, V. Honavar, and L. Miller. Intelligent agents for intrusion detection, 1998.
- [HWHM00]: G. Helmer, J. Wong, V. Honavar, and L. Miller. Lightweight agents for intrusion detection, 2000.
- [JMKM00] :W. Jansen, P. Mell, T. Karygiannis, and D. Marks. Mobile agents in intrusion detection and response, 2000.

- [Can98]: J. Cannady. Artificial neural networks for misuse detection, 1998.
- [FH]: Stephanie Forrest and Steven A. Hofmeyr. Immunology as information processing.
- [FHS97]: Stephane Forrest, Steven A. Hofmeyr, and Anil Somayaji. Computer immunology. *Communications of the ACM*, 40(10) :88–96, 1997.
- [HF00] :Steven A. Hofmeyr and Stephanie Forrest. Architecture for an artificial immune system. *Evolutionary Computation*, 8(4) :443–473, 2000.
- [Clo] : Jacques Clot. *La réponse imunitaire chez l'homme*. Documentation immunologique INAVA.

Résumé

Le développement technologique, l'utilisation d'Internet à grande échelle et l'augmentation des moyens de stockage et de l'échange d'information sont contribué à un nombre élevé de cyber-attaques, qui exploitent les failles des systèmes d'information et les points faibles des systèmes de protection contre les intrusions, ce qui rend le processus de sécurisation de ces informations très important. Les spécialistes du domaine de la sécurité informatique s'occupent d'élaborer les outils nécessaires pour assurer la sécurité de l'information en développant de nouvelles technologies basées sur des moyens d'intelligence artificielle afin de détecter les pénétrations non découvertes par des dispositifs ordinaires. Dans ce contexte, nous visons à travers ce travail à réaliser un modèle de détection d'intrusions qui s'appuis sur l'apprentissage automatique des informations échangées et de signaler les divergences par rapport à ce fonctionnement de référence comme des attaques .en se basant sur les réseaux de neurones artificiels, et la base de données de référence KDDCUP99 pour générer et évaluer le modèle proposé

Mots clés : intrusions, classification, réseaux de neurones, intelligence artificiel, KDDCUP99, apprentissage automatique.

Abstract

The large-scale technological, development and use of the Internet and the increase in the means of storage and exchange of information have contributed to a high number of cyber attacks, which exploit the gaps in information systems and the weak points of protection systems against intrusions, which makes the process of securing this information very important. The security specialists are making the necessary tools to ensure information security by developing new technologies based on artificial intelligence to detect the penetrations undiscovered by ordinary devices. In this context, we aim, through this work, to realize an intrusion detection model based on the Machine learning functioning of the exchanged information and to signal the the divergences from this reference functioning as attacks. Based on the artificial neural networks, and the KDDCUP99 reference database to generate and evaluate the proposed model.

Keywords: intrusions, classification, neural networks, artificial intelligence, KDDCUP99, Machine learning.

ملخص:

لقد أدى التطور التكنولوجي ، استعمال شبكة الانترنت على نطاق واسع و تعدد وسائل تخزين و تبادل المعلومات إلى ارتفاع عدد الهجمات الالكترونية و عمليات قرصنة البيانات من خلال استعمال ثغرات الأنظمة المعلوماتية و نقاط ضعف انظمه الحماية من الاختراقات مما جعل عملية تامين المعلومات تكتسي أهمية كبيرة و من اجل مواكبة هذه التطورات عمل المختصين في محال امن المعلوماتية على توفير الأدوات و الوسائل اللازمة لتامين المعلومات بوضع تقنيات جديدة تعتمد على الذكاء الاصطناعي لتحديد الاختراقات الغير مكتشفة بواسطة أجهزة تامين المعلومات العادية وفي هذا الإطار نهدف من خلال هذا العمل الى انجاز نموذج لكشف

الاختراقات يعتمد على فهم طبيعة البيانات العادية و تصنيف الغير معروفة كاختراقات و هذا باستعمال الحركات العصبية الاصطناعية كما اعتمدنا على قاعدة المعطيات KDDCup99كمرجع لانجاز و تقييم النموذج المقترح
كلمات مفتاحيه: اختراقات ، تصنيف ، شبكات عصبية ، ذكاء اصطناعي ،.KDDCup99