République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université 20 Aout 1955-SKIKDA

Faculté des Sciences Département d'Informatique

Mémoire de fin d'études en vue de l'obtention du diplôme de

Master en Informatique

Option : Systèmes Informatiques (SI)

Une étude de techniques de classification et de clustering en détection d'intrusion.

Réalisé par :

- CHOUMAH Rayenne
- ALI BOUAITA Nour

Encadre par :

- Dr. NAFIR Abdenacer
- Dr. MAZOUZI Smaine

Année universitaire 2O21/2O22

REMERCIEMENTS

Louange tout d'abords à dieu qui nous a donné la force pour terminer ce modeste travail. Toutes nos infinies gratitudes à nos encadreurs MM. NAFIR ABDENACER et MAZOUZI SMAINE pour leur encadrement et leurs aides précieuses.

Nous remercions aussi les membres de jury qui nous ont fait l'honneur d'accepter le jugement de notre travail. Notre sincère reconnaissance à nos enseignants du département : **informatique**.

Enfin nous remercions les travailleurs de GNL1K qui nous ont aidés et ils **WASSIM**, **SOFIANE**, **MARWA** et **Mr BOUDROUMA.** Et tous nos amis, trouve ici l'expression de notre profonde gratitudes et respects.

Dédicace

Je dédie se modeste travail à ma lumière blanche dans le noir, ma boussole quand j'ai perdu, mais comment j'ai perdu quand ton amour qui me trace le chemin.

Ma chère mère **« NADIA »**et ma deuxième maman **« NOURA »** pour tous ce que tu fais pour moi, pour ton sacrifice et pour ton amour.

A ma grande mère bien aimée que dieu m'a aidée grâce à sa prière constante.

A tous mes chers frères **« MOHAMED »** et **« AMINE »**et à ma sœur **« CHAIMA »**.

A mon ami intime **« ABDOU »** ❤️

A mes amies proches **« KHAOULA, NOUR, MANAR, INES, AYA »**.

A mon voisin **« BRAHIM »** 🙂

A tous qui m'ont aidé à accomplir ce travail

Enfin je dédie ce travail à mais longues année d'étude.

Merci beaucoup 🙂

RAYENNE

Dédicace

Je dédie cette mémoire :

*A ma chère **Mami***

*A mon cher **Papi***

*A mes chères **grands-mères** et mes chers **grands-pères***

Qui n'ont jamais cessé de formuler des prières à mon égard de me soutenir et de m'épauler pour que je puisse atteindre mes objectifs.

*A mes chers **oncles***

*A mes chères **tantes***

Source d'espoir et de motivation

A toute ma famille

*A ma chère amie intime **Hanane***

*A mon amie **Hanna***

Qui m'ont aidé à supporter mes moments difficiles

*A mon Binôme **Rayane***

Pour son entente et sa sympathie

*A toutes les amies : **Inès, Kawtar, Aya, Manar***

A tous les bébés et les enfants de la famille

(Ismail, Karam, Djoud)

Maria, Fatima, Yousra, Rania

Abderrahmane, Abdelhak, Mennou, Idris, Amir, Siradj, Daidou, Rida

Source de joie et de bonheur

*Et a tous qui aime **Nour***

Résumé / ملخص / Abstract

Résumé :

Les réseaux informatiques sont exposés aux plusieurs types d'attaques c'est pour cela nous avons besoins des moyens pour les protéger. Parmi ces moyens nous citons les systèmes de détection d'intrusions. Cependant, avec l'évolution qui a touché les techniques d'attaques, ces systèmes ne donnent plus de bons résultats. Dans ce mémoire, nous avons mené une étude expérimentale pour décider quel est le classifieur le plus approprié aux données de la base d'intrusions KDD. Nous avons considéré quatre classifieurs à savoir le naive bayes, l'arbre de décision (C4.5), arbre aléatoire, k-plus proche voisins (Knn), et le cluster k means .et nous avons comparé leurs précisions et leurs temps d'exécution.

Mots Clés : sécurité informatique, les méthodes d'apprentissages automatiques, système de détection d'intrusion.

ملخص

تتعرض شبكات الكمبيوتر لعدة أنواع من الهجمات، ولهذا نحتاج إلى وسائل لحمايتها. من بين هذه الوسائل نستشهد بأنظمة كشف التسلل. ومع ذلك، مع التطور الذي أثر على تقنيات الهجوم، لم تعد هذه الأنظمة تعطي نتائج جيدة. في هذه الرسالة، أجرينا دراسة تجريبية لتحديد المصنف الأنسب لبيانات قاعدة بيانات اقتحام KDD. لقد درسنا أربع مصنفات وهي الخلايا الساذجة وشجرة القرار (C4.5) والشجرة العشوائية وأقرب جيران (Knn) والوسائل العنقودية وقارننا دقتها وتنفيذ أوقات تشغيلها..

الكلمات المفتاحية: أمن الحاسوب، طرق التعلم الآلي، نظام كشف التسلل، مجموعة المصنفات

Abstract:

Computer networks are exposed to several types of attacks, which is why we need the means to protect them. Among these means we cite intrusion detection systems. However, with the evolution that has affected attack techniques, these systems no longer give good results. In this thesis, we conducted an experimental study to decide which is the most appropriate classifier for the data of the KDD intrusion database. We considered four classifiers namely the naive bayes, the decision tree (C4.5), random tree, k-nearest neighbors (Knn), and the cluster k means .and we compared their accuracies and their running times execution.

Keywords: computer security, machine learning methods, intrusion detection system, set of classifiers.

Sommaire

Sommaire

Sommaire

Sommaire

Sommaire

La liste des Figures

La liste des Figures

La liste des Figures

La liste des Tableaux

Introduction Générale

Introduction générale :

Le développement remarquable du domaine des nouvelles technologies de l'information et de la communication (NTIC) ces dernières années, et l'utilisation de l'outil informatique à grande échelle, plus l'accessibilité du réseau internet par un grand nombre d'utilisateurs avec leurs différentes intentions qui peuvent être parfois destructifs, cela rend les données sensibles ainsi que les ressources des utilisateurs et des sociétés vulnérables au vol, ou exploités pour des raisons malveillantes. Face à toutes ces menaces, la sécurité optimale des systèmes informatiques et des réseaux est devenue un enjeu stratégique et pour assurer cette sécurité, différents outils ont été utilisés, tels que les pares-feux et les anti-virus.

Malheureusement les systèmes antivirus ou les firewalls sont la plupart du temps inefficaces face à ces nouvelles menaces sophistiquées, dont la propagation peut s'avérer extrêmement rapide. C'est pour pallier ce manque que sont apparus récemment de nouvelles solutions de sécurité appelées systèmes de détection des intrusions (IDS) qui consistent à examiner le trafic réseau, collecter tous les événements, les analyser et générer des alertes en cas d'identification de tentatives malveillantes. Ces systèmes sont devenus jour après jour très utilisés dans les stratégies de sécurité des réseaux et systèmes informatiques. Néanmoins, le défi majeur pour les systèmes de détection d'intrusions réside dans leur capacité à déterminer tout comportement malicieux que ce soit de l'intérieur ou de l'extérieur du système informatique.

En effet, le domaine de la détection d'intrusion est très ouvert à la recherche et au développement, où des produits logiciels et des solutions pratiques commencent à apparaitre jour après jour et qui fonctionnent selon deux principaux modes : l'approche par scénario et l'approche comportementale.

L'approche par scénario, basée sur la comparaison du comportement d'utilisation du système avec des signatures d'attaques connues préalablement et ne permet pas la détection des nouvelles attaques sans mise à jour de la base de signatures ce qui représente un inconvénient majeur de cette approche.

Par contre, l'approche comportementale consiste à construire un modèle identifiant les comportements déviants du modèle comportemental normal. Ce modèle est le résultat d'une phase d'apprentissage sur une grande base de données et son avantage principal est la possibilité de détecter de nouvelles attaques.

Ce mémoire est organisé comme suit :

Le chapitre 1 est consacré à la sécurité informatique et la détection d'intrusion. Le chapitre présente essentiellement les différents types d'attaques contre les systèmes informatiques, et aussi les différentes mesures de protection et cela utilisant Intrusion Detection Systems . Après avoir présenté l'intérêt des IDS, et les différentes architectures selon lesquelles sont construit. Il a été introduit également les IDS collaboratifs.

Le chapitre 2 : nous avons présenté l'état de l'art de l'apprentissage automatique. Tout d'abord, nous donnons un bref aperçu historique de la prémisse à l'essor du domaine. Ensuite, nous donnons ses différents types et quelques algorithmes utilisés.

Le chapitre 3 : présente notre protocole expérimental des deux classifieurs considérés. Dans ce chapitre, nous avons donné un aperçu des classifieurs supervisés, et nous avons détaillé les quatre classifieurs et le cluster que nous allons utiliser.

Le chapitre 4 : présente les outils logiciels utilisés, les éléments d'implémentation, et quelques résultats expérimentaux.

Chapitre 01 : Sécurité d’informatique et détection d'intrusion

Introduction :

Dans le premier chapitre de notre mémoire de master, nous présentons brièvement les dernières technologies liées à la sécurité informatique, notamment la détection d'intrusion. Nous commençons par présenter les différents types d'attaques qui peuvent cibler à la fois les applications et les réseaux, ainsi que les données. Ensuite, nous introduisons les principales méthodes et techniques principales de défense contre différentes attaques.

I. Sécurité d'informatique :

La sécurité informatique est un ensemble de moyens mis en œuvre pour réduire la vulnérabilité d'un système aux menaces accidentelles ou intentionnelles. Il s'agit d'empêcher les actions non autorisées des utilisateurs des systèmes informatiques afin d'assurer la confidentialité, l'intégrité, la disponibilité et la traçabilité des informations traitées. [1]

1. Attaques et sécurité informatique :

Un individu ou un groupe de personnes peut attaquer l'ordinateur d'un individu ou d'un groupe de personnes morales ou physiques. La dépendance humaine contemporaine aux réseaux informatiques inspire non seulement le goût de la réussite, mais aussi la volonté d'acquérir de l'argent, des convictions politiques, des objectifs militaires et de la curiosité. La défaillance du système peut être due à un manque de budget, de temps d'installation, de personnel qualifié, de politique de sécurité, de protection efficace sur le marché. Aujourd'hui, cependant, le coût d'une attaque et de sa résolution peut être très élevé. C'est pourquoi les entreprises et toutes les organisations sont très intéressées par la sécurité informatique.

L'idée intuitive de la sécurité informatique est de limiter l'accès à un système informatique avec une sécurité parfaite, les informations ne peuvent jamais être compromises car les utilisateurs non autorisés ne peuvent jamais y accéder. Cependant, une sécurité parfaite n'est pas réaliste. C'est pourquoi nous essaierons de prévenir, détecter et répondre aux attaques afin que la même attaque ne se reproduise plus. La prévention peut limiter les circonstances dans lesquelles une attaque se produit.

Pour ce faire, nous utilisons des techniques d'authentification, de cryptage et même de camouflage pour convaincre les attaquants potentiels que le programme n'en vaut pas la peine.

Comme la prévention n'est pas parfaite, la détection peut identifier certaines fonctionnalités qui violent la politique de sécurité.

2. Objectifs de sécurité informatique :

Afin d'être efficaces et rentables, les entreprises d'aujourd'hui communiquent avec leurs filiales, leurs partenaires, voire fournissent des services aux particuliers, ce qui a conduit à une énorme ouverture de l'information. Avec un réseau ouvert, la sécurité devient un facteur décisif dans le fonctionnement normal d'une entreprise ou d'une organisation.

Il s'agit toujours d'une entreprise ou d'une organisation qui détient certaines informations qui ne doivent être divulguées qu'à un certain nombre de personnes, ou ne doivent pas être modifiées, ou doivent être mises à la disposition des utilisateurs de manière transparente. Ces informations sont ciblées si et seulement s'il existe une menace et que les systèmes protégeant ces informations sont vulnérables.

Pour cette raison, nous nous référons à la sécurité de l'information comme un état de protection, face à un risque identifié, qui résulte de toutes les mesures générales et particulières prises pour assurer la confidentialité, l'intégrité et la disponibilité des informations traitées, où :

Authentification : L'authentification vise à vérifier l'identité du processus de communication. Plusieurs solutions simples sont mises en œuvre pour cela, comme l'utilisation d'un nom d'utilisateur et d'un mot de passe.

Autorisation : informations permettant d'identifier et d'autoriser les utilisateurs identifiés et autorisés à accéder aux ressources de l'entreprise et les actions autorisées sur ces ressources. Cela couvre toutes les ressources de l'entreprise.

Confidentialité : un ensemble de mécanismes qui permettent aux communications de données de rester privées entre l'expéditeur et le destinataire. La cryptographie ou le cryptage des données est la seule solution fiable pour assurer la confidentialité de ses données.

Intégrité : Un mécanisme pour s'assurer que l'information n'a pas été modifiée par des personnes non autorisées.

Disponibilité : Ensemble des mécanismes permettant d'assurer l'accessibilité des ressources de l'entreprise, impliquant l'architecture du réseau, la bande passante, les plans de sauvegarde, etc.

Non-répudiation : Un mécanisme pour s'assurer qu'un message a été envoyé par l'expéditeur et reçu par le destinataire.

Traçabilité : Un mécanisme pour suivre les opérations effectuées sur les ressources de l'entreprise. Cela suppose que tous les événements d'application sont archivés pour une enquête ultérieure.

3. Graphique d'attaque :

Une attaque est une tentative de violation de l'un des objectifs de sécurité informatique, et une intrusion est une attaque réussie.

L'attaque peut se résumer en six points : [1].

- Recueillir des informations sur le système.
- Intrusion dans le système à cause de ces informations.
- Installer des systèmes qui permettent de futures ré-intrusions, comme l'insertion de code dans l'EEPROM.
- Rechercher la propagation d'une intrusion dans un autre système, permettant ainsi une attaque distribuée.
- Plantages du système.
- Effacer les traces des agresseurs.

4. Type d'attaque :

Il existe essentiellement 04 types d'attaques :

4.1. Attaques réseau :

Ce type d'attaque repose principalement sur des failles liées au protocole ou à sa mise en œuvre. Dans ce qui suit, nous présenterons quelques attaques bien connues.

- **Technologie de numérisation :**

L'analyse de port est une méthode permettant de déterminer les types d'attaques pouvant être lancées sur une machine cible. Cette technique comprend la communication d'informations sur les machines analysées, en particulier le système d'exploitation et les services installés. En conséquence, nous pouvons identifier avec précision les vulnérabilités de sécurité et donc les types d'attaques possibles sur les machines concernées.[2].

- **Usurpation IP :**

Usurpation d'adresse IP pour faire croire que la demande provient d'une machine autorisée. Une bonne configuration des routeurs d'entrée peut empêcher les machines externes de se faire passer pour des machines internes [2].

- **Usurpation DNS :**

Cette technique est utilisée pour inciter les serveurs DNS à accepter les intrus. La solution est de séparer le DNS pour le LAN du DNS pour l'espace public.

- **Flooding :**

Utilisé pour déconnecter quelqu'un de son adresse IP, en utilisant un programme appelé flooder. Le flooder envoie un ping à la victime. ping est utilisé pour calculer la vitesse à laquelle vous pouvez communiquer avec une autre machine sur le réseau. Tout ping générera une réponse. En augmentant le nombre de pings, le serveur déconnectera la victime car elle enverra trop de données en réponse au flood.

- **Smurf :**

Le smurf ou "reflection attack" est une technique basée sur l'utilisation de serveurs de diffusion pour faire tomber le réseau.

- **Web bug :**

Le courrier publicitaire est envoyé en HTML, apparemment normal, avec une image transparente difficile à voir en raison de sa taille. Si le message est ouvert lors du processus de connexion, la demande de téléchargement d'image confirme la lecture du message et la validité de votre adresse.

- **Hoax (rumeur) :**

Un « canular » est une rumeur diffusée par e-mail. Ces rumeurs colportaient souvent de prétendus problèmes de sécurité découverts par des services officiels ou bien connus. Ils peuvent engorger le réseau, causant un réel préjudice à certaines entreprises. Avant de retransmettre un tel message, il est prudent de vérifier son authenticité.

- **Hackers et crackers :**

Hackers : dans les premières expériences d'ARPA-net, il y avait une communauté, une culture partagée, de programmeurs expérimentés et d'experts en réseau. Les membres de cette culture ont inventé le terme "hacker". Ces informaticiens sont généralement discrets, anti-autoritaires et animés par la curiosité.

Cracker : principalement quelqu'un qui s'introduit à distance dans un système informatique et exécute un logiciel qui nécessite une licence mais n'a pas besoin d'être acheté, en utilisant généralement des outils écrits par d'autres et trouvés sur Internet.

4.2. Attaque applicative :

Les attaques applicatives se basent sur des failles dans les programmes utilisés, voire sur des erreurs de configuration. Cependant, comme auparavant, ces attaques peuvent être classées en fonction de leur origine.

➢ Problèmes de configuration :

Il est rare qu'un administrateur réseau configure correctement un programme. Généralement, ils se contentent d'utiliser la configuration par défaut. Ceux-ci sont généralement dangereux pour faciliter le fonctionnement du logiciel. De plus, des erreurs peuvent se produire lors de la configuration du logiciel. Une mauvaise configuration du serveur peut entraîner l'accès à des fichiers importants ou compromettre l'intégrité du système d'exploitation

➢ Injection SQL :

L'injection SQL exploite des paramètres d'entrée non validés. Comme son nom l'indique, le but de l'injection SQL est d'injecter du code SQL dans une requête de base de données afin que les informations puissent être extraites de la base de données ou même détruites [2].

➢ Intermédiaire :

L'un des piratages les plus sophistiqués qu'un utilisateur non autorisé peut effectuer est ce que l'on appelle une attaque de type "man-in-the-middle". Il s'agit d'une attaque visant à récupérer des données sensibles en transit sur un réseau local. L'attaque impliquait trois machines : le serveur cible, le poste client et la machine de l'attaquant.

Le but de cette attaque est d'intercepter la communication de la machine de l'attaquant entre le serveur cible et le poste client sans que les entités concernées puissent suspecter que le canal de communication a été compromis [3].

Figure1.1 : Man in the middle.

4.3. Déni de service : Comme mentionné ci-dessus, un déni de service est une attaque conçue pour rendre un service indisponible. Cela peut se faire de plusieurs manières : en surchargeant le réseau afin que la machine soit complètement inaccessible ; ou en plantant à distance l'application en tant qu'application.

L'utilisation d'un buffer overflow peut planter une application à distance.

En raison de certaines instructions malveillantes et d'erreurs de programmation, des personnes malveillantes peuvent rendre indisponibles des services (serveurs Web, serveurs de messagerie, etc.) voire des systèmes entiers.

Voici quelques cyberattaques connues qui peuvent rendre les services indisponibles [2] :

SYN Flooding : Utilisez la connexion en trois étapes de TCP (prise de contact à trois voies : SYN/SYN-ACK/ACK). Le principe est de faire attendre un grand nombre de connexions TCP. L'attaquant envoie de nombreuses requêtes de connexion (SYN), reçoit des SYN-ACK, mais ne répond jamais avec des ACK. Les connexions en cours consomment des ressources mémoire et peuvent entraîner une saturation et des plantages du système.

 UDP Flooding : le trafic UDP a priorité sur TCP. L'objectif est donc d'envoyer beaucoup de paquets UDP, ce qui consommera toute la bande passante, rendant toutes les connexions TCP indisponibles.

Exemple : faites une demande de chargement (port 19/service de génération de caractères) vers une machine en usurpant l'adresse source et le port à rediriger vers l'écho (port de service de la chaîne reçue à plusieurs reprises) à partir d'une autre machine.

 Packet Fragment : Utilise une mauvaise gestion de la défragmentation au niveau ICMP. Exemple : La paix de la mort. La quantité de données est supérieure à la taille maximale d'un paquet IP.

Une fois cela fait, il ne reste plus qu'à donner l'ordre d'attaquer toutes les machines d'un coup, afin de répliquer l'attaque par milliers. Ainsi, une simple attaque comme SYN Flooding peut rendre une machine ou un réseau complètement inaccessible.

4.4. Attaque de données (contenu) :

Les données transmises par les protocoles applicatifs (contenu) peuvent constituer une menace pour l'intégrité des systèmes qui les reçoivent. Les principales attaques de ce type que nous retrouvons : Virus, Vers, Applets Java, Trojans... spécifiés par un code malveillant ou malware.

4.4.1. Virus :

Un virus est un programme informatique malveillant conçu et écrit pour se reproduire. Cette capacité à s'auto-répliquer peut affecter votre ordinateur sans votre permission et à votre ins. En termes plus techniques, un virus classique s'attache à l'un de vos programmes exécutables et se copie systématiquement sur tout autre programme exécutable que vous lancez. Les virus informatiques ne surviennent pas spontanément. Ils doivent être écrits dans un but précis. En plus de se répliquer, un virus peut ou non se comporter de manière plus ou moins nocive, depuis l'affichage d'un simple message jusqu'à la destruction de toutes les données [3].

4.4.2. Vers :

Un ver est un programme parasite. Il n'a pas besoin d'être auto-propagé. Son but est de consommer des ressources système : CPU, mémoire, espace disque, bande passante. Cette applet dépend du système d'exploitation ou du logiciel. Comme toutes les données binaires, elles transitent par disquette, CD ROM, réseau (LAN ou WAN). Depuis la démocratisation des virus (notamment en raison de la prolifération des générateurs de virus), le nombre de nouveaux vers a chuté de manière spectaculaire. Cependant, il y en a encore. Pour les prévenir, les mêmes stratégies que celles employées contre les virus sont souvent utilisées [3].

4.4.3. Cheval de Troie :

Un cheval de Troie est un programme qui prétend faire quelque chose mais qui fait autre chose. Leur nom vient du célèbre cheval de Troie de la Grèce antique, offert en cadeau, mais qui a en fait reçu le cheval dans le but de dévaster et de détruire la ville.

Un cheval de Troie sur un ordinateur est un programme exécutable autonome qui se comporte avec une action spécifique. Cependant, lorsque ce programme est lancé, il peut, par exemple, formater le disque dur, voler des mots de passe ou envoyer des informations confidentielles au créateur via Internet [3].

4.4.4. Bombe logique :

Une bombe logique est une partie d'un programme qui reste inactive dans le système hôte jusqu'à ce qu'un certain moment ou événement se produise ou que certaines conditions soient remplies pour déclencher un effet destructeur dans celui-ci [3].

4.4.5. Porte arrière :

Les portes dérobées (backdoors) sont des logiciels de communication cachés, tels qu'installés par des virus ou des chevaux de Troie, qui permettent à des attaquants externes d'accéder au réseau des ordinateurs des victimes [4].

4.4.6. Spyware :

Un logiciel espion (également appelé logiciel espion ou logiciel espion) est un logiciel malveillant installé sur un ordinateur ou un autre appareil mobile dans le but de collecter et de transmettre des informations sur l'environnement dans lequel il est installé. Généralement à l'insu de l'utilisateur. L'essor de ce type de logiciel est lié à l'essor d'Internet comme moyen de transmission de données [5].

4.4.7. Pourriel :

Le spam, le courrier indésirable ou le courrier indésirable est une communication électronique non sollicitée, principalement par courrier électronique. Il s'agit généralement d'envois en nombre à des fins publicitaires [6].

5. Outils de sécurité : nous présentons ci-dessous un ensemble non exhaustif d'outils de sécurité :

5.1. Antivirus :

Un logiciel antivirus est un logiciel conçu pour identifier, neutraliser et éliminer les logiciels malveillants (dont les virus ne sont qu'un exemple). Ceux-ci peuvent être basés sur des exploits de failles de sécurité, mais peuvent également être des programmes qui modifient ou suppriment des fichiers, qu'il s'agisse de documents destinés à l'utilisateur d'un ordinateur infecté ou de fichiers nécessaires au bon fonctionnement de l'ordinateur.

L'antivirus vérifie les fichiers et les e-mails, les secteurs de démarrage (pour détecter les virus de démarrage), et vérifie également la RAM de votre ordinateur, les supports amovibles (clés USB, CD, DVD, etc.), les données transférées sur n'importe quel réseau (y compris Internet), et plus encore. [7].

5.2. Pare-feu :

Un pare-feu est une solution matérielle ou logicielle mise en œuvre dans une infrastructure réseau pour filtrer l'accès à des ressources réseau définies. Il n'autorise l'accès qu'aux utilisateurs autorisés munis d'une clé ou d'un badge et crée une couche de protection entre le réseau et le monde extérieur. Il a des filtres intégrés qui empêchent les documents non autorisés ou potentiellement dangereux d'entrer dans le système. Il enregistre également les tentatives d'intrusion dans les journaux envoyés aux administrateurs réseau. Il contrôle également l'accès aux applications et empêche les transferts d'usage [8].

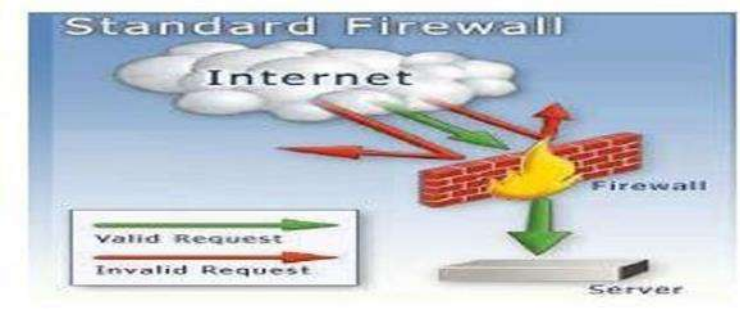


Figure1.2 : Pare-feu (Firewall)**.**

5.3. Cryptographie :

La cryptographie est l'étude des méthodes de transmission de données de manière sécurisée. Pour protéger le message, il est transformé de façon à ce qu'il soit incompréhensible ; c'est ce

qu'on appelle le cryptage, et il donne le texte chiffré à partir du texte en clair. En revanche, le décryptage est l'acte de reconstruire le texte en clair à partir du texte chiffré en utilisant une clé et un algorithme de décryptage spécifiques [7].

5.4. Réseau privé virtuel (VPN) :

Les réseaux privés virtuels (VPN) permettent aux utilisateurs de créer des chemins virtuels sécurisés entre les sources et les destinations. Avec la croissance d'internet, il est intéressant de permettre ce type de processus de transfert de données sûr et sécurisé. Grâce au principe du tunnel, chaque extrémité est identifiée et les données y transitent après avoir été chiffrées.

Un des gros intérêts des VPN est de créer des réseaux privés à moindre coût. En cryptant les données, tout se passe comme s'il s'agissait d'une connexion établie en dehors d'Internet. En revanche, le Web doit être pris en compte, car la qualité de service (QoS) ne peut être garantie. Le principe du VPN repose sur la technologie tunnel. Cela inclut la création d'un chemin virtuel après avoir déterminé l'expéditeur et le destinataire. La source crypte ensuite les données et les achemine le long de ce chemin virtuel. Les données à transférer peuvent appartenir à un protocole autre qu'IP. Dans ce cas, le protocole de tunneling encapsule les données en ajoutant des en-têtes. Autoriser le routage des trames dans le tunnel. Le tunneling est un ensemble de processus d'encapsulation, de transmission et de décapsulation [9].

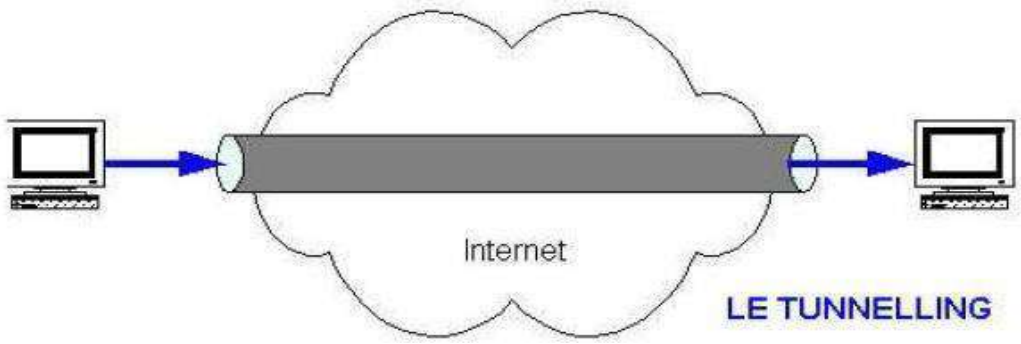


Figure1.3 : Réseau privé virtuel (VPN)**.**

5.5. Système de détection d'intrusion (IDS) :

Un Système de Détection d'Intrusion (ou IDS : Intrusion Detection System) est un mécanisme destiné à détecter une activité anormale ou suspecte sur une cible d'analyse (réseau ou hôte). Par conséquent, il peut apprendre les tentatives d'intrusion réussies et échouées [10].

II. Détection d'intrusion :

Aujourd'hui, les attaques se produisent plus rapidement que jamais et tout le monde risque de perdre des données importantes. Malheureusement, les systèmes antivirus ou pare-feu sont pour la plupart inefficaces contre ces nouvelles menaces. C'est pour pallier cette déficience que de nouveaux composants de sécurité appelés systèmes de détection d'intrusion ont récemment vu le jour.

Ceux-ci permettent de dissuader les cyberattaques anticipées en générant des alertes, des avertissements en présence de menaces externes ou internes, ce qui contribue à réduire le temps et les efforts fournis par les administrateurs car la surveillance peut se faire sans surveillance humaine. La protection des données clients contre les intrusions vous permet de gagner la confiance de vos clients et partenaires et de maintenir une solide réputation au sein de votre organisation.

1. la définition de la détection d'intrusion :

La détection d'intrusion est le processus de surveillance des événements dans un système informatique ou réseau et de leur analyse pour détecter les signes d'intrusion, et est définie comme une tentative de compromettre la confidentialité, l'intégrité, la disponibilité ou le contournement des mécanismes de sécurité d'un ordinateur ou d'un réseau.

Les intrusions sont causées par des attaques accédant à des systèmes via Internet, des utilisateurs système autorisés tentant d'obtenir des privilèges supplémentaires auxquels ils ne sont pas autorisés et des utilisateurs autorisés abusant de privilèges donnés. Les systèmes de détection d'intrusion sont des logiciels ou du matériel qui automatisent le processus de surveillance et d'analyse [11].

2. Type d'IDss :

Les pirates utilisent diverses méthodes d'attaque, certaines exploitent les vulnérabilités du réseau et d'autres exploitent les vulnérabilités de la programmation. C'est pourquoi la détection d'intrusion doit se faire à plusieurs niveaux. Nous détaillerons donc ci-dessous les principales caractéristiques des trois IDS

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

Host-Based Intrusion Detection Host Base IDS (HIDS) surveille le trafic sur une seule machine.

Il analyse les journaux système, les appels et enfin vérifie l'intégrité des fichiers. HIDS nécessite un système sain pour vérifier l'intégrité des données. Si le système est piraté, HIDS ne fonctionnera plus.

2.2. Système de détection d'intrusion « réseau » (NIDS) :

Un NIDS est un IDS utilisé pour protéger un réseau. Ils incluent généralement une sonde (telle qu'une machine) qui écoute et surveille tout le trafic réseau en temps réel, puis analyse et génère des alertes lorsqu'une intrusion ou un paquet apparemment dangereux est détecté.

2.3. Système de détection d'intrusion « hybride » :

L'IDS hybride combine les caractéristiques du NIDS et du HIDS. Ils permettent de surveiller les réseaux et les terminaux. Les sondes sont placées à des points stratégiques et agissent comme NIDS et/ou HIDS selon leur emplacement. Toutes ces sondes envoient ensuite des alertes à une machine qui va tout centraliser et relier les informations provenant de multiples sources.

Par conséquent, nous comprenons que l'IDS hybride est basé sur une architecture distribuée, où chaque composant unifie son format d'envoi. Cela facilite la communication et l'extraction d'alertes plus précises.

3. L'architecture fonctionnelle d'IDS :

Nous décrivons dans cette section les trois composants qui composent généralement un système de détection d'intrusion.

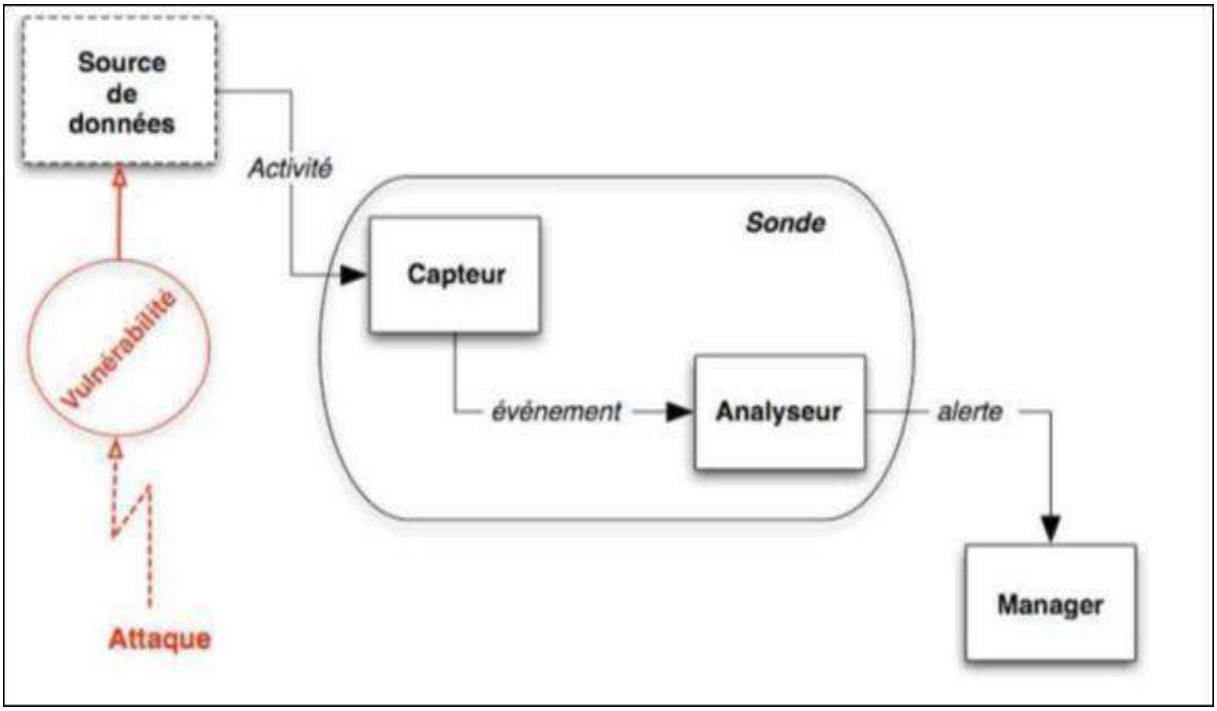


Figure1.4 : illustre les interactions entre ces trois composants.

3.1. Capteur :

Le capteur observe l'activité du système à travers la source de données et fournit à l'analyseur une série d'événements qui renseignent sur l'évolution de l'état du système.

Les capteurs peuvent se contenter de transmettre directement ces données brutes, mais effectuent généralement un prétraitement. Pour cela, nous distinguons trois types de capteurs en fonction des sources de données utilisées pour observer l'activité du système : les capteurs système, les capteurs réseau et les capteurs applicatifs.

3.2. Analyseur :

Le but de l'analyseur est de déterminer si le flux d'événements fourni par le capteur contient des caractéristiques d'activité malveillante.

3.3. Gestionnaire :

Le gestionnaire collecte les alertes générées par les capteurs, les met en forme et les présente à l'opérateur. En définitive, le gestionnaire est responsable de la réponse à l'adoption, qui peut être :

- Isoler l'attaque, visant à limiter l'impact de l'attaque
- Suppression d'attaque, essayant d'arrêter une attaque.
- Récupération, c'est-à-dire l'étape au cours de laquelle le système revient à un état sain.
- Diagnostic, la phase d'identification du problème.

4. Caractéristiques de l'ID :

Les propriétés souhaitées d'un système de détection d'intrusion sont [13] :

- Capable de fonctionner en continu avec une intervention humaine minimale.
- Il est difficile pour un attaquant de désactiver ou de modifier sa configuration.
- Capacité à se contrôler et à détecter s'il vient d'être manipulé par un attaquant.
- Utilisation minimale des ressources (calcul, stockage, etc.) sur le système sur lequel il est installé.
- Capacité à accepter les mises à jour et les changements de configuration pour refléter les nouvelles dispositions de la politique de sécurité et les changements qui peuvent survenir dans l'organisation (nouvelles acquisitions, réorganisations, etc.).
- Facilité de déploiement : Installation facile et portabilité de la configuration, etc.
- Interopérabilité avec d'autres systèmes et outils de sécurité informatique.
- Il doit être tolérant aux pannes, c'est-à-dire qu'il doit pouvoir retrouver son état de fonctionnement initial après un crash causé par une manipulation accidentelle ou l'activité d'une personne non malveillante.
- À mesure que le nombre de systèmes à surveiller augmente, tout comme le potentiel d'attaques, on peut s'attendre à ce que l'IDS ait les caractéristiques suivantes :
  - Il doit être en mesure de superviser un grand nombre de sites tout en fournissant des résultats rapides et précis.
  - Il doit assurer un « service de crise minimum », c'est-à-dire que si certains composants de l'IDS cessent de fonctionner, d'autres composants doivent être le moins possible affectés par cet état dégradé.

5. Classification des systèmes de détection d'intrusion :

Les différents systèmes de détection d'intrusion disponibles peuvent être classés selon plusieurs critères [12] :

- Méthode de détection.
- Comportement du système après détection.

- Sources de données.
- Fréquence d'utilisation.

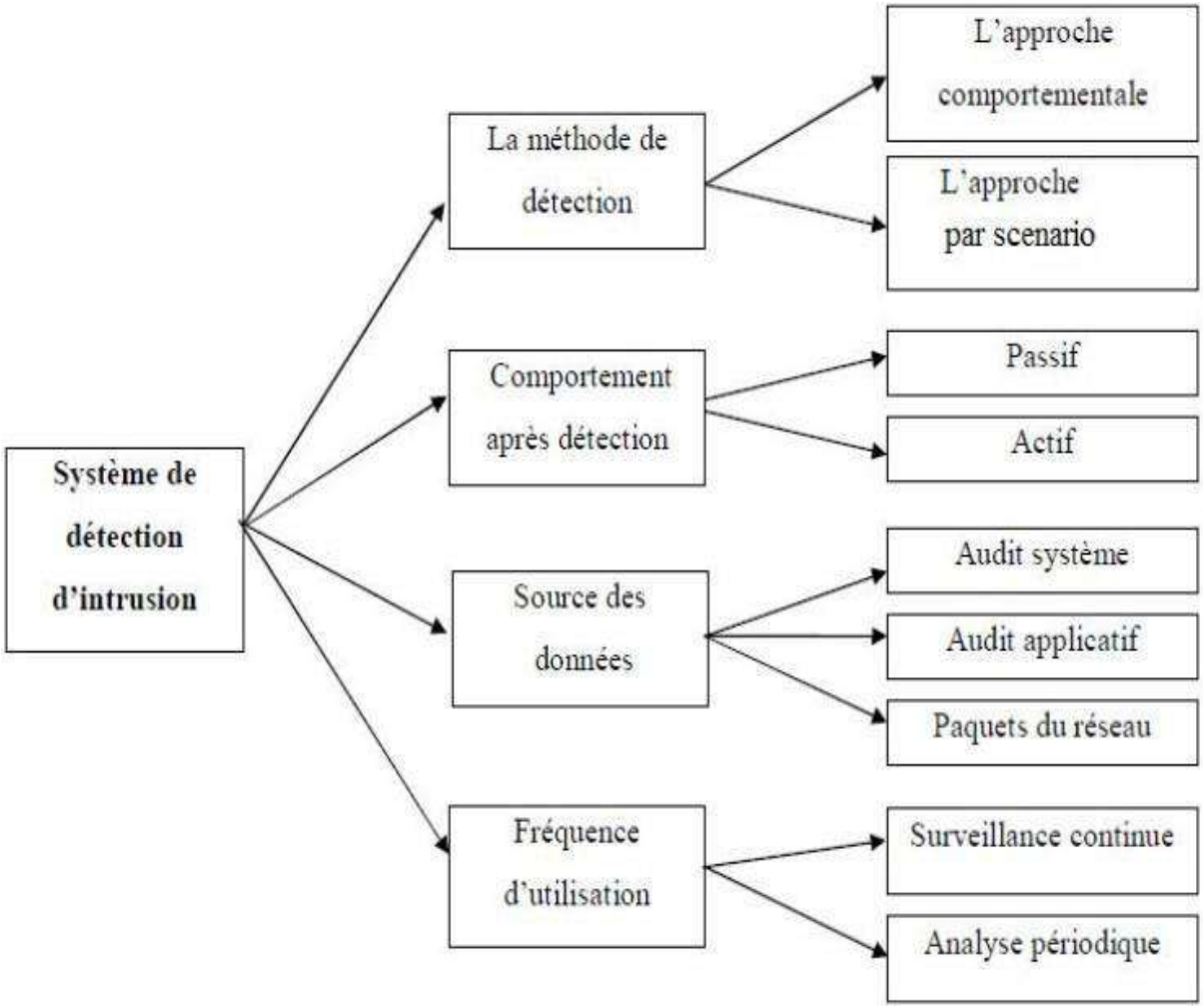


Figure1.5 : Classification d'un système de détection d'intrusions.

5.1. Méthode de détection IDS :

Il existe deux méthodes de détection

5.1.1. Méthode de comportement :

Cette technique consiste à détecter les intrusions en fonction du comportement d'un utilisateur ou d'une application, c'est-à-dire à créer un modèle basé sur le comportement habituel du système et à surveiller tout écart par rapport à ce comportement [14].

Plusieurs paramètres sont possibles : charge CPU, quantité de données échangées, durée et temps de connexions aux ressources, protocole utilisé et répartition statistique des applications, etc.

5.1.2. Par scène ou méthode signée :

La technique repose sur la connaissance des techniques utilisées par les attaquants contenus dans la base de données, qui compare l'activité des utilisateurs dans la base de données, puis déclenche une alerte lorsqu'un événement se produit qui dépasse le profil.

5.2. Comportement après détection d'intrusion :

Il existe deux types de réponses selon l'IDS utilisé. Les réponses réactives s'appliquent à tous les IDS, des réponses plus ou moins proactives sont mises en place.

5.2.1. Réponse passive :

Lorsqu'une attaque est détectée, le système d'intrusion ne prend aucune mesure, il génère simplement une alerte à l'administrateur système sous la forme d'une alerte lisible avec des informations sur chaque attaque. Une réponse passive entraîne généralement une reconfiguration automatique du pare-feu pour bloquer les adresses IP source impliquées dans l'intrusion. Cependant, si un pirate parvient à accéder à une adresse IP sensible, comme l'accès à un routeur ou à un serveur DNS, les entreprises qui mettent en œuvre une reconfiguration systématique de leurs pare-feux risquent d'être coupées du monde extérieur.

5.2.2. Réponse active :

Une réponse proactive est une attaque à réponse directe qui implique des actions automatisées prises par l'IDS pour couper rapidement les connexions suspectes lorsque le système détecte une intrusion. Par exemple, interrompez le processus de l'attaque, puis empêchez le prochain accès de l'attaquant.

5.3. La nature des données d'analyse :

La nature des données analysées comprend :

5.3.1. Audit système :

Les audits système sont générés par le système d'exploitation de l'hôte. Ces données permettent à IDS de surveiller l'activité des utilisateurs sur l'hôte.

5.3.2. Examen de la candidature :

Les données à analyser sont générées directement par l'application, comme les fichiers journaux générés par les serveurs FTP et les serveurs Web. L'avantage de cette catégorie est que les données produites sont très complètes, informatives et en quantité modérée.

Ces types d'informations sont généralement intégrés dans les IDS basés sur l'hôte.

5.3.3. Sources d'informations sur le réseau :

Ce sont des données de trafic réseau. Cette source d'information est prometteuse car elle permet de collecter et d'analyser les paquets circulant sur le réseau. Un IDS qui utilise ces sources de données est appelé : NIDS Network-based IDS.

5.4. Fréquence d'utilisation :

La fréquence d'utilisation d'un système de détection d'intrusion peut exister sous deux formes :

5.4.1. Suivi régulier :

Ce type de système de détection d'intrusion analyse périodiquement diverses sources de données pour détecter d'éventuelles intrusions ou des anomalies passées.

5.4.2. Surveillance en temps réel :

Les systèmes de détection d'intrusion en temps réel se concentrent sur le traitement et l'analyse en continu des informations générées par différentes sources de données. Il limite les dommages causés par l'attaque car il permet de prendre des mesures pour réduire la progression de l'attaque détectée.

6. Attaques contre les IDS :

Étant donné que les IDS sont une méthode de protection d'un système, les attaquants attaquent souvent le système qu'il protège avant de l'attaquer, et comme les IDS sont des systèmes informatiques, ils contiennent des vulnérabilités.

Il existe plusieurs types d'attaques telles que le déni de service, l'insertion, l'évasion et la modification des paquets transmis du capteur à l'analyseur.

Déni de service :

Nous désactivons l'IDS en le saturant d'informations. Pour résoudre ce problème, il faut bien filtrer et stocker les informations, et disposer d'un IDS efficace pour gérer un ensemble de paquets sans perte. De plus, le système de surveillance peut vérifier.

L'effet réel de l'insertion d'IDS :

Nous insérons des paquets supplémentaires, l'IDS les acceptera et la machine les rejettera.

Par conséquent, en les combinant, le message final dans la machine cible peut être différent de celui dans l'IDS.

Un exemple d'une telle attaque consiste à insérer de mauvaises sommes de contrôle dans des paquets, que la machine cible rejettera et que l'IDS acceptera. En fait, pour des raisons de performances, IDS ne les rejettera pas, même s'ils ont de mauvaises sommes de contrôle.

Échapper :

IDS ignore ces paquets, mais le système attaqué les considère.

Éditer :

Nous modifions les paquets envoyés du capteur à l'analyseur. Il faut donc veiller à leur protection.

Par conséquent, des techniques de chiffrement contemporaines telles que TLS (Transport Layer Security) peuvent être utilisées.

Les raisons de ce type d'attaque sont diverses : raisons de performance, ambiguïté des standards protocolaires, méconnaissance des standards protocolaires par les fabricants d'IDS.

Détection IDS :

Tout comme attaquer une machine, pour attaquer un IDS, vous devez être capable de le détecter. Voici quelques exemples :

Usurpation d'adresse MAC (contrôle d'accès au support) : NIDS met l'interface de capture de paquets réseau en mode promiscuité et ils peuvent voir tous les paquets qui passent. Ainsi en envoyant un paquet ICMP (Internet Control Message Protocol) de type "echorequest", la machine qui le reçoit doit envoyer un paquet ICMP de type "echoreply" contenant la trinité des adresses MAC présentes dans le réseau, on peut vérifier si la machine est la réponse. Ainsi, le NIDS étant en mode promiscuité, il voit le paquet et renvoie une écho-réponse sans même vérifier qu'il est bien le destinataire du paquet ICMP.

Mesure de latence : les NIDS étant en mode promiscuité, ils doivent gérer tous les paquets réseau. Donc, si une machine devient de plus en plus lente à répondre après avoir envoyé beaucoup de paquets à toutes les machines, nous pouvons supposer qu'elle est en mode promiscuité et donc probablement NIDS.

Demande d'observation : Après une attaque, l'IDS envoie généralement un message à l'ordinateur central, qui gérera toutes les alertes. Donc en regardant les paquets, on peut essayer de trouver le centre.

L'un des problèmes de ce type d'attaque est que, dans la plupart des cas, l'IDS n'avertit pas l'ensemble du système qu'il est en panne. Ceci est encore plus dangereux lorsque l'IDS ne sait pas s'il fonctionne correctement.

De cette façon, après avoir attaqué l'IDS, les pirates peuvent attaquer le système en toute impunité. Par conséquent, un IDS doit être capable de distinguer si un pirate attaque le système d'un IDS ou une machine protégée par IDS pour potentiellement bloquer le système [1].

7. Conclusion :

Parmi les nombreuses technologies de défense informatique, IDS a sa place. En effet, ils permettent l'analyse du trafic, seul moyen de détecter d'éventuelles attaques sur le système sans connaissance préalable. Nous avons vu que les IDS varient en fonction de différentes normes de conception et d'utilisation. Dans notre étude, nous avons cherché à améliorer la précision de l'analyseur IDS en sélectionnant correctement le meilleur classificateur en fonction des données d'apprentissage disponibles. Dans le chapitre suivant, nous passerons en revue les méthodes de classification liées à l'apprentissage automatique.

Chapitre 02 : apprentissage automatique

Introduction :

L'apprentissage automatique est un sous-domaine de l'informatique qui s'intéresse à la construction d'algorithmes utiles, en s'appuyant sur un ensemble d'exemples d'un phénomène particulier. Ces exemples peuvent provenir de la nature, être créés par l'homme ou générés par un autre algorithme. L'apprentissage automatique peut également être défini comme le processus de résolution de problèmes du monde réel en 1) collectant un ensemble de données et 2) en construisant de manière algorithmique un modèle statistique basé sur cet ensemble de données. Ce modèle statistique devrait en quelque sorte être utilisé pour résoudre des problèmes réels.

Une définition un peu plus générale : « L'apprentissage automatique est une branche de la science qui donne aux ordinateurs la capacité d'apprendre sans programmation explicite » [16]. Et une autre, plus technique : « on dit qu'un programme informatique apprend de l'expérience E par rapport à tâche T et la mesure de performance P, si sa performance sur T, mesurée par P, s'améliore avec l'expérience de E » [17].

Dans ce chapitre, nous avons présenté l'état de l'art de l'apprentissage automatique. Tout, nous donnons un bref aperçu historique de la prémisse à l'essor du domaine. Ensuite, nous donnons ses différents types et quelques algorithmes utilisés.

1. La renaissance de l'apprentissage automatique :

En 2006, Geoffrey Hinton et al. Ont publié un article [18] montrant comment former un réseau neuronal profond capable de reconnaître des chiffres manuscrits avec une précision de pointe (> 98%). Ils appellent cette technique "l'apprentissage en profondeur". La formation de réseaux de neurones profonds était largement considérée comme impossible à l'époque, et la plupart des chercheurs ont abandonné l'idée depuis les années 1990. Le nouvel article montre que l'apprentissage en profondeur est non seulement possible, mais fournit une correspondance incroyable de résultats qu'aucune autre technique d'apprentissage automatique ne peut égaler (grâce à des quantités massives de puissance de calcul et de grandes quantités de données). Cet enthousiasme s'est rapidement propagé à de nombreux autres domaines de l'apprentissage automatique.

Depuis 10 ans, le machine learning a conquis l'industrie : il est désormais au cœur de la magie des produits high-tech capables de classer les résultats de recherche sur le Web, de prendre en charge la reconnaissance vocale des smartphones et de recommander des vidéos, de battre des champions du monde au jeu de Go, et Suite.

2.types d'apprentissage automatique :

Il existe de nombreux types de systèmes d'apprentissage automatique. Dans ce qui suit, nous les classons selon qu'ils nécessitent ou non une supervision humaine (apprentissage supervisé, non supervisé, semi-supervisé et par renforcement). [19]

2.1 Apprentissage supervisé :

Dans l'apprentissage supervisé, les données d'apprentissage fournies à l'algorithme comprennent des solutions, appelées étiquettes.

Une tâche typique d'apprentissage supervisé est la classification. Un filtre anti-spam en est un bon exemple : il se compose de nombreux exemples d'e-mails et de leurs catégories (spam ou non), et il doit apprendre à classer les nouveaux e-mails, comme présenté dans la figure 2.1.

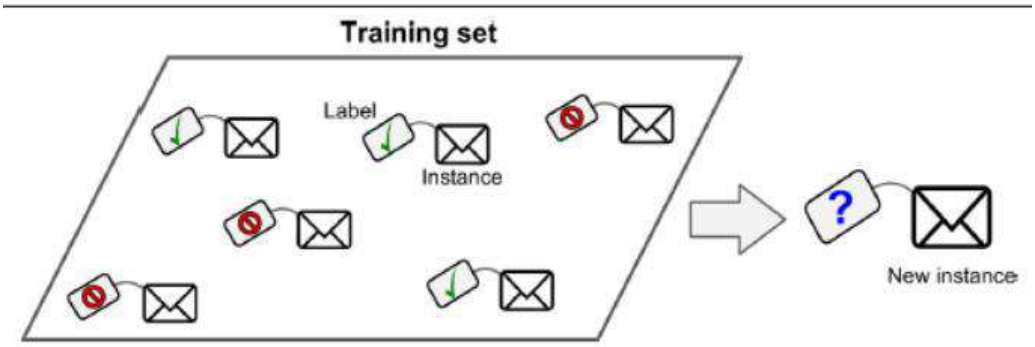


Figure 2.1 : Ensemble de formation étiquetée pour l'apprentissage supervisé (par exemple, classification des spams) [19].

Une autre tâche typique consiste à prédire une valeur cible, telle que le prix d'une voiture, sur la base d'un ensemble de caractéristiques appelées prédicteurs (kilométrage, âge, marque, etc.). Ce type de tâche est appelé régression. Pour former ce système, nous devons lui donner de nombreux exemples de voitures, y compris leurs prédicteurs et leurs étiquettes (c'est-à-dire leurs prix). La figure 2.2 décrit une tâche de régression [19]

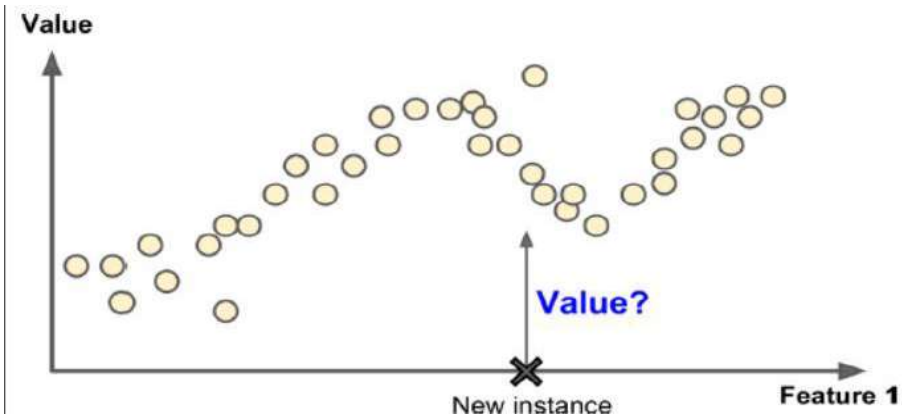


Figure 2.2 : Régression [19].

Notez que certains algorithmes de régression peuvent également être utilisés pour la classification et vice versa. Par exemple, la régression logistique est souvent utilisée pour la classification car elle produit une valeur qui correspond à la probabilité d'appartenir à une classe donnée (par exemple, 20 % de chance d'être un spam).

Voici quelques-uns des algorithmes d'apprentissage supervisé les plus importants :

·K plus proches voisins.

· Régression linéaire.

· régression logistique.

· Machines à vecteurs de support (SVM).

· Arbres de décision et forêts aléatoires.

2.2 Apprentissage non supervisé :

Dans l'apprentissage non supervisé, les données d'apprentissage ne sont pas étiquetées. Le système essaie d'apprendre sans enseignant.

Le modèle n'a pas de « réponses » à tirer ; il doit donner un sens aux données en termes d'observations elles-mêmes.

L'apprentissage non supervisé nous permet d'aborder des problèmes avec peu ou pas de connaissances sur ce à quoi nos résultats devraient ressembler. Nous pouvons obtenir une structure à partir de données pour lesquelles nous ne connaissons pas nécessairement l'influence des variables.

Voici quelques-uns des algorithmes d'apprentissage non supervisés les plus importants :

· Clustering

K-Means, Analyse des clusters hiérarchiques (HCA), Maximisation des attentes.

. Visualisation et réduction de la dimensionnalité :

– Analyse en composantes principales (ACP).
– Kernel PCA.
– L'encastrement linéaire local (LLE).
– T-distribué Stochastic Neighbor Embedding (t-SNE).

. Apprentissage des règles d'association :

– Apriori.

– Eclat.

2.2.1 Clustering :

Par exemple, disons que nous avons beaucoup de données sur les visiteurs du blog. Nous pourrions utiliser un algorithme de regroupement pour essayer de détecter des groupes de visiteurs similaires. L'algorithme de clustering ne dit jamais à quel groupe appartient un visiteur : il trouve ces connexions sans notre aide. Par exemple, l'algorithme peut remarquer que 40 % de vos visiteurs sont des hommes qui aiment les bandes dessinées et lisent généralement votre blog la nuit, tandis que 20 % sont de jeunes fans de science-fiction qui visitent le site pendant la semaine. -fin etc Si nous utilisons un algorithme de clustering hiérarchique, cet algorithme peut subdiviser chaque groupe en groupes plus petits. Cela peut aider à localiser les messages pour chaque groupe. [20]
Dans la figure 2.3 ci-dessous, nous montrons comment un ensemble de points peut être classé en trois sous-ensembles :

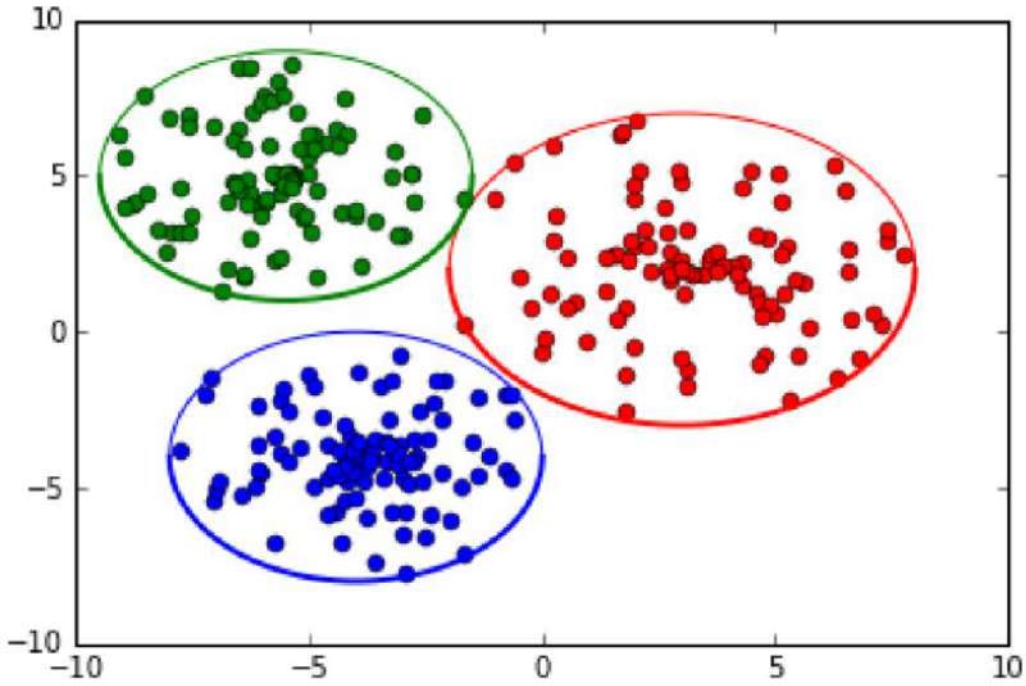


Figure 2. 3 : Clustering [20] .

2.2.1.1 K-means

K-means est un algorithme non supervisé de clustering non hiérarchique. Il permet de regrouper en clusters distincts les observations du data set. Ainsi les données similaires se retrouveront dans un même cluster. Par ailleurs, une observation ne peut se retrouver que dans un cluster à la fois

(Exclusivité d'appartenance). Une même observation, ne pourra donc, appartenir à deux clusters différents.

Le clustering est une méthode d'apprentissage non supervisé (unsupervised learning). Ainsi, on n'essaie pas d'apprendre une relation de corrélation entre un ensemble de features d'une observation et une valeur à prédire, comme c'est le cas pour l'apprentissage supervisé. L'apprentissage non supervisé va plutôt trouver des patterns dans les données. Notamment, en regroupant les choses qui se ressemblent.

En apprentissage non supervisé, les données sont représentées comme suit :

$$X=\begin{pmatrix} x(1,1) & x(1,2)\ldots\ldots..x(1,n) \\ x(2,1) & x(2,2)\ldots\ldots..x(2,n) \\ \ldots & \ldots \\ x(m,1) & (m,2)\ldots.x(m,n) \end{pmatrix}$$

Chaque ligne représente un individu (une observation). A l'issu de l'application du clustering, on retrouvera ces données regroupées par ressemblance. Le clustering va regrouper en plusieurs familles (clusters) les individus/objets en fonction de leurs caractéristiques. Ainsi, les individus se trouvant dans un même cluster sont similaires et les données se trouvant dans un autre cluster ne le sont pas.

Il existe deux types de clustering :

- Le clustering hiérarchique
- Le clustering non-hiérarchique (partitionnement)

le concept de clustering tel que je l'ai décrit lors de ce paragraphe est un clustering non hiérarchique.

Pour pouvoir regrouper un jeu de données en cluster distincts, l'algorithme K-Means a besoin d'un moyen de comparer le degré de similarité entre les différentes observations. Ainsi, deux données qui se ressemblent, auront une distance de dissimilarité réduite, alors que deux objets différents auront une distance de séparation plus grande.

Les littératures mathématiques et statistiques regorgent de définitions de distance, les plus connues pour les cas de clustering sont :

- **La distance Euclidienne** : C'est la distance géométrique qu'on apprend au collège. Soit une matrice à variables quantitatives. Dans l'espace vectoriel .La distance euclidienne entre deux observations et se calcule comme suit :

$$d(x_1, x_2) = \sqrt{\sum_{j=1}^{n} (x_{1n} - x_{2n})^2}$$

- **La distance de Manhattan (taxi-distance)** : est la distance entre deux points parcourus par un taxi lorsqu'il se déplace dans une ville où les rues sont agencées

selon un réseau ou un quadrillage. Un taxichemin est le trajet fait par un taxi lorsqu'il se déplace d'un nœud du réseau à un autre en utilisant les déplacements horizontaux et verticaux du réseau.

2.2.2 Réduction des dimensions

L'objectif est de simplifier vos données sans perdre trop d'informations. Une façon de le faire est de combiner plusieurs traits corrélés en un seul. Par exemple, le kilométrage d'une voiture peut être fortement corrélé à son âge, donc un algorithme de réduction dimensionnelle les combinera en une seule caractéristique qui représente l'usure de la voiture. C'est ce qu'on appelle l'extraction de caractéristiques.

2.2.3 Détection des anomalies :

La détection d'anomalies est un domaine passionnant dont l'objectif est d'identifier les objets distants qui s'écartent de la distribution globale des données. La détection des valeurs aberrantes s'est avérée nécessaire dans de nombreux domaines, par exemple, la détection des transactions
Anormales par carte de crédit pour prévenir la fraude, la détection des défauts de fabrication ou la suppression automatique des valeurs aberrantes de l'ensemble de données avant Transférez-les vers un autre algorithme d'apprentissage. Le système est formé avec des instances normales, et lorsqu'il voit une nouvelle instance, il peut dire si elle ressemble à une instance normale ou s'il peut s'agir d'une anomalie, comme illustré dans la figure ci-dessous. Figure 2.4 » :

Figure 2.4 : Détection d'anomalies.

2.2.4 Règles d'association d'apprentissage :

L'objectif est d'exploiter de grandes quantités de données et de découvrir des relations intéressantes entre les attributs. Par exemple, disons que nous possédons un supermarché. L'application des règles d'association aux registres des ventes pourrait révéler que les personnes qui achètent de la sauce barbecue et des pommes de terre ont également tendance à acheter du steak. Il est donc peut-être possible de placer ces éléments à proximité les uns des autres.

2.3 Apprentissage semi-supervisé :

Les algorithmes d'apprentissage semi-supervisé peuvent traiter des données d'apprentissage partiellement étiquetées, généralement Beaucoup de données non étiquetées et une petite quantité de données étiquetées, et faire des prédictions pour tous les points invisibles. L'apprentissage semi-supervisé est courant lorsque des données non étiquetées sont facilement disponibles mais que les étiquettes sont coûteuses à acquérir. Différents types de problèmes qui surviennent dans les applications, y compris les tâches de classification, de régression ou de classement, peuvent être présentés comme des exemples d'apprentissage semi-supervisé. L'espoir est de distribuer des données non étiquetées accessibles.
Les algorithmes peuvent l'aider à atteindre de meilleures performances que l'apprentissage supervisé. L'analyse des conditions pour atteindre cet objectif fait l'objet de nombreuses recherches théoriques et appliquées en apprentissage automatique moderne [21].
Certains services d'hébergement de photos, tels que Google Photos, en sont de bons exemples. Lors du téléchargement de photos de famille sur le service, il reconnaît automatiquement que la même personne A apparaît sur les photos 1, 5 et 11, tandis qu'une autre personne B apparaît sur les photos 2, 5 et 7. C'est la partie non supervisée de l'algorithme (clustering). Tout ce dont le système a besoin maintenant, c'est que nous lui disions qui sont ces personnes. Un seul tag par personne et la possibilité de nommer tout le monde sur chaque photo sont utiles pour la recherche de photos.

La plupart des algorithmes d'apprentissage semi-supervisé sont une combinaison d'algorithmes non supervisés et supervisés. Par exemple, les Deep Belief Networks (DBN) sont basés sur des composants non supervisés appelés Restricted Boltzmann Machines (RBM), qui sont empilés les uns sur les autres. Les RBM sont formés séquentiellement et sans

supervision, puis des techniques sont utilisées pour affiner l'ensemble du système, puis le système entier est réglé avec précision grâce à des techniques d'apprentissage supervisées.

2.4 Apprentissage par renforcement :

Un scénario général d'apprentissage par renforcement est illustré à la figure 2.5. Contrairement au scénario d'apprentissage supervisé, l'apprenant ici ne reçoit pas passivement un ensemble de données étiquetées. Au lieu de cela, il recueille des informations à travers le cours de l'action qui interagit avec l'environnement. En réponse à une action, un apprenant ou un agent reçoit deux types d'informations : son état actuel dans l'environnement et des récompenses à valeur réelle spécifiques à la tâche et à son objectif correspondant. L'objectif de l'agent est de maximiser sa récompense, il détermine donc le meilleur plan d'action ou la meilleure stratégie pour atteindre cet objectif.

Cependant, les informations qu'il obtient de l'environnement ne sont qu'une récompense directe liée à l'action qu'il vient d'effectuer. Un aspect important de l'apprentissage par renforcement consiste à envisager des récompenses ou des pénalités différées.8
Les agents sont confrontés à un dilemme entre l'exploration d'états et d'actions inconnus pour obtenir plus d'informations sur l'environnement et les récompenses, et l'optimisation de leurs récompenses avec les informations déjà collectées. Ceci est connu comme le compromis entre l'exploration et l'exploitation, et il est lié à l'apprentissage par renforcement [21].

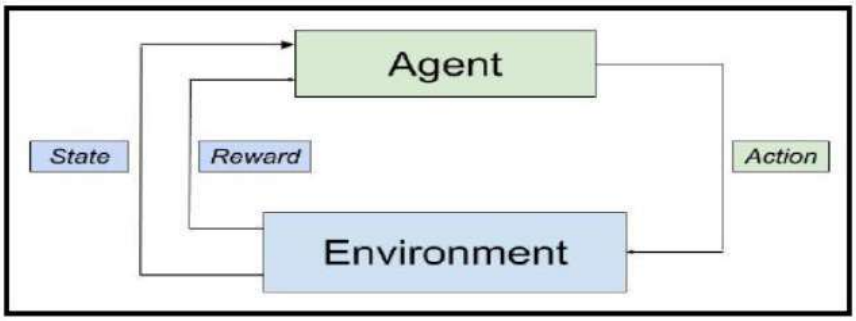


Figure 2.5 : Apprentissage par renforcement.

3.Algorithmes d’apprentissage supervisé :

3.1 Les k-plus proches voisins :

L'algorithme k-plus proche voisin (k-NN) est basé sur l'ensemble des données. En effet, pour les observations que nous voulons prédire et qui ne font pas partie des données, l'algorithme va rechercher les k instances les plus proches de notre observation, et pour chaque observation sélectionner la classe majoritaire parmi ses k plus proches voisins.

La méthode k-NN est une technique d'apprentissage supervisé qui est considérée comme l'une des méthodes les plus simples dans le domaine de la classification. Il permet de classer de nouvelles observations (vecteurs d'entités extraites) en calculant la distance à partir des données d'apprentissage, et en prenant les k plus proches voisins (en fonction de la distance). Ensuite, observez la classe la plus courante dans les k plus proches voisins et affectez cette classe à une nouvelle observation.

Le programme AlphaGo de DeepMind est un bon exemple d'apprentissage par renforcement : il a fait la une des journaux en mars 2016 lorsqu'il a battu le champion du monde Lee Sedol au jeu de Go. Il apprend sa stratégie gagnante en analysant des millions de jeux, puis joue de nombreux jeux contre lui-même.

Bien que le temps d'apprentissage de l'algorithme k-NN soit court, le temps de requête réel (et l'espace de stockage) peut être plus long que d'autres modèles. Cela est particulièrement vrai lorsque le nombre de points de données augmente, car toutes les données d'apprentissage doivent être conservées, pas seulement l'algorithme.
Le plus gros inconvénient de cette approche est qu'elle peut être boguée par des propriétés non pertinentes qui cachent des propriétés importantes. Il existe des moyens de corriger cela, comme l'application de pondérations aux données. Comme nous l'avons détaillé ci-dessus, l'algorithme k-NN calcule la distance entre les points de données. Pour cela, nous utilisons la formule de distance euclidienne :

$$đ(p,q)=đ(q,p)$$

$$=\sqrt{(q1-p1)^2+(q2-p2)^2+\cdots+(qn-pn)^2}$$

$$đ(p,q)=đ(q,p)=\sqrt{\sum_{i=1}^{n}(qi-pi)^2}$$

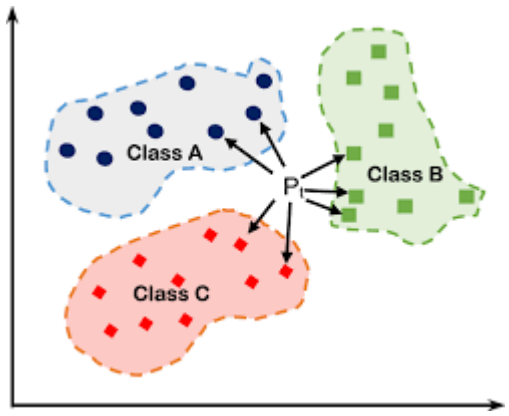


Figure 2.6 : KNN.

3.2 Bayes naïfs :

Si les données ne sont pas complexes et que la tâche est relativement simple, l'algorithme Naïve Bayes peut être utilisé.

Il s'agit d'un classificateur qui surpasse la régression logistique et les k plus proches voisins lorsqu'il utilise une quantité limitée de données pour former le modèle.

Naive Bayes est également un bon choix lorsque les ressources CPU et mémoire sont un facteur limitant. Parce que c'est si simple, cela ne provoque pas de surcharge de données et peut s'entraîner très rapidement. Il fonctionne également sur de nouvelles données continues utilisées pour mettre à jour le classifieur. D'autres classificateurs peuvent être meilleurs si la taille et la variance des données augmentent et que vous avez besoin d'un modèle plus complexe. De plus, sa simple analyse n'est pas une bonne base pour des hypothèses complexes.

Naive Bayes est souvent le premier algorithme que les scientifiques essaient d'analyser du texte. Il s'agit d'un algorithme de classification qui applique une estimation de densité aux données. L'algorithme utilise le théorème de Bayes et suppose que les données prédites sont conditionnellement indépendantes. C'est un classifieur probabiliste basé sur le théorème de Bayes :

$$P(A/B) = \frac{\mathrm{P(B/A)\ .P(A)}}{P(B)}$$

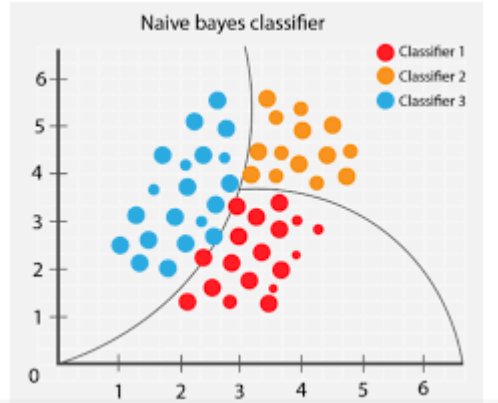


Figure 2.7 : Naive Bayes.

3.3 Régression linéaire :

La régression linéaire représente une méthode d'apprentissage supervisé très simple. En particulier, la régression linéaire est un outil utile pour prédire les réponses quantitatives.

Bien que cela puisse sembler un peu ennuyeux par rapport à certaines des méthodes d'apprentissage statistique les plus modernes, la régression linéaire reste une méthode d'apprentissage statistique utile et largement utilisée. De plus, c'est un bon point de départ pour de nouvelles méthodes qui sont considérées comme des généralisations ou des extensions de la régression linéaire [22].

3.3.1 Régression linéaire simple :

La régression linéaire simple porte bien son nom : il s'agit d'une méthode linéaire très simple de prédiction d'une réponse quantitative Y basée sur une seule variable prédictive X. Il suppose une relation approximativement linéaire entre X et Y. Mathématiquement, on peut écrire cette relation linéaire comme suit :
g ≈ b0 + b1c (4.1.1)

"≈" peut être lu comme "approximativement modélisé comme". Nous le décrivons parfois en disant que nous régressons g sur c (ou régressons g sur c).
Par exemple, c pourrait représenter le montant dépensé pour les publicités télévisées et g pourrait représenter les ventes. On peut alors régresser les ventes de téléviseurs en ajustant le modèle.

Ventes ≈ b 0 + b 1 × TV.
Dans les équations précédentes, b 0 et b 1 sont deux constantes inconnues qui représentent les termes d'ordonnée à l'origine et de pente dans le modèle linéaire. Ensemble b 0 et b 1 sont appelés les coefficients ou paramètres du modèle. Une fois que nous avons utilisé nos données d'entraînement pour générer des estimations pour les coefficients du modèle b 0' et b 1', nous pouvons prédire les ventes futures en calculant des valeurs spécifiques basées sur les dépenses publicitaires télévisées :

g' = b 0' + b' x,

où g' représente la prédiction de g basée sur c = x. On utilise ici une notation, ‘, représente une estimation du coefficient du paramètre inconnu, ou la valeur prédite de la réponse. En fait, b 0

et b 1 sont inconnus. Par conséquent, avant de pouvoir utiliser (4.1.1) pour faire des prédictions, nous devons utiliser les données pour estimer les coefficients.

Oui :

(x1, y1), (x2, y2), . . . , (xn, yn)

Représente n paires d'observations, chacune constituée d'une mesure de c et d'une mesure de g. Dans l'exemple publicitaire, l'ensemble de données inclut les budgets publicitaires télévisés et les ventes de produits. Notre objectif est d'obtenir des estimations des coefficients b 0' et b 1' afin que le modèle linéaire (4.1.1) s'ajuste bien aux données disponibles, c'est-à-dire g' ≈ b 0' + b 1'xi pour i =1, . . .non. En d'autres termes, nous voulons trouver une intersection b 0' et une pente b 1' telles que la ligne résultante soit aussi proche que possible des points de données.
Il existe plusieurs façons de mesurer la proximité. Cependant, la méthode la plus courante consiste à minimiser le critère des moindres carrés. Un exemple comme le montre la Figure 2.8 :

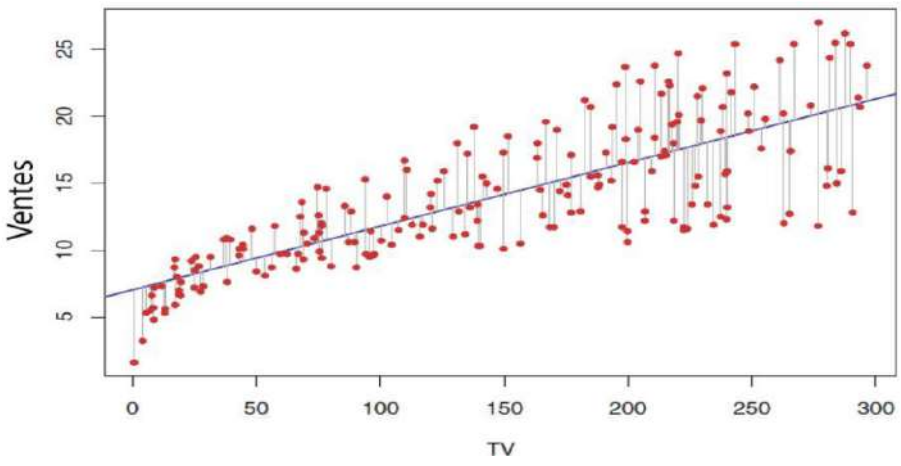


Figure 2 :8 : Régression linéaire [3].

Pour les données publicitaires, les moindres carrés correspondent à une régression des ventes sur les budgets publicitaires de la télévision (TV). Les ajustements sont obtenus en minimisant la somme des carrés.

Chaque segment de ligne grise représente une erreur et l'ajustement est compromis en faisant la moyenne de leurs carrés. Dans ce cas, l'ajustement linéaire capture l'essence de la relation, bien qu'il soit insuffisant sur le côté gauche du graphique.

Soit gi' = b0' + b1'xi la prédiction de g basée sur la ième valeur de c. Alors ei = gi-gi' représente le ième résidu - c'est la différence entre la ième réponse observée et la ième réponse prédite par notre modèle linéaire. Nous définissons la somme résiduelle des carrés (RSS) comme suit :

$$RSS = e^2_1 + e^2_2 + ... + e^2_n,$$

Ou de manière équivalente à :

$$RSS = (g_1 - b_0' - b_1'x1)^2 + (y2 - b_0' - b_1'x_2)^2 + ...+(g_n - b_0' - b_1'x_n)^2.$$

3.3.2 Régression linéaire multiple :

La régression linéaire simple est une méthode utile pour prédire une réponse basée sur une seule variable prédictive. En pratique, cependant, nous avons souvent plus d'un prédicteur. Par exemple, dans les données publicitaires, nous avons examiné la relation entre les ventes et la publicité télévisée. Nous disposons également de données sur les dépenses consacrées aux publicités à la radio et dans les journaux, et nous pourrions être intéressés de savoir si l'un de ces supports est en corrélation avec les ventes. Comment pouvons-nous étendre notre analyse des données publicitaires pour tenir compte de ces deux prédicteurs supplémentaires ? Une option consiste à exécuter trois régressions linéaires simples distinctes, chacune utilisant un support publicitaire différent comme prédicteur.
Cependant, la méthode d'ajustement d'un modèle de régression linéaire simple séparé pour chaque indicateur n'est pas entièrement satisfaisante. Premièrement, étant donné les niveaux des trois budgets publicitaires, il n'est pas clair comment une prévision de ventes unique serait faite, puisque chaque budget est associé à une équation de régression distincte.
Deuxièmement, chacune des trois équations de régression ignore les deux autres médias pour former des estimations des coefficients de régression.

Nous avons rapidement découvert que si les budgets des médias étaient corrélés dans notre ensemble de données, cela pourrait conduire à des estimations très trompeuses.
Plutôt que d'ajuster un modèle de régression linéaire simple séparé pour chaque prédicteur, une meilleure approche consiste à étendre le modèle de régression linéaire simple afin qu'il puisse s'adapter directement à plusieurs prédicteurs. Nous pouvons le faire en donnant à chaque prédicteur un coefficient de pente distinct dans un modèle unique. D'une manière générale, supposons que nous ayons p différents prédicteurs.
Le modèle de régression linéaire multiple prend alors la forme :
g = b0 + b1X1 + b2X2 + ::: + bpXp + e ;
Comme dans le cas de la régression linéaire simple, les coefficients de régression b0, b1, ... bp sont inconnus et doivent être estimés. Étant donné les estimations b0', b1', . . . bp', nous pouvons faire des prédictions en utilisant la formule suivante :
g' = b0' + b1'X1 + b2'X2 + ::: + pb'Xp ;
Les paramètres sont estimés à l'aide de la même méthode des moindres carrés que nous avons vue dans la régression linéaire simple. On choisit b0, b1, ... bp pour minimiser la somme des carrés résiduelle :
RSS = ån= (gi -g′)2
moi 1 moi
= ån= (gi -b ' -b 'xi1 -b 'xi2 -::: -bp'xi p)2
je 1012
La figure 2.9 montre un cadre tridimensionnel avec deux prédicteurs et une variable de réponse, avec la droite de régression des moindres carrés transformée en plan. Choisissez le plan pour minimiser la somme des carrés des distances verticales entre chaque observation (en rouge) et le plan :

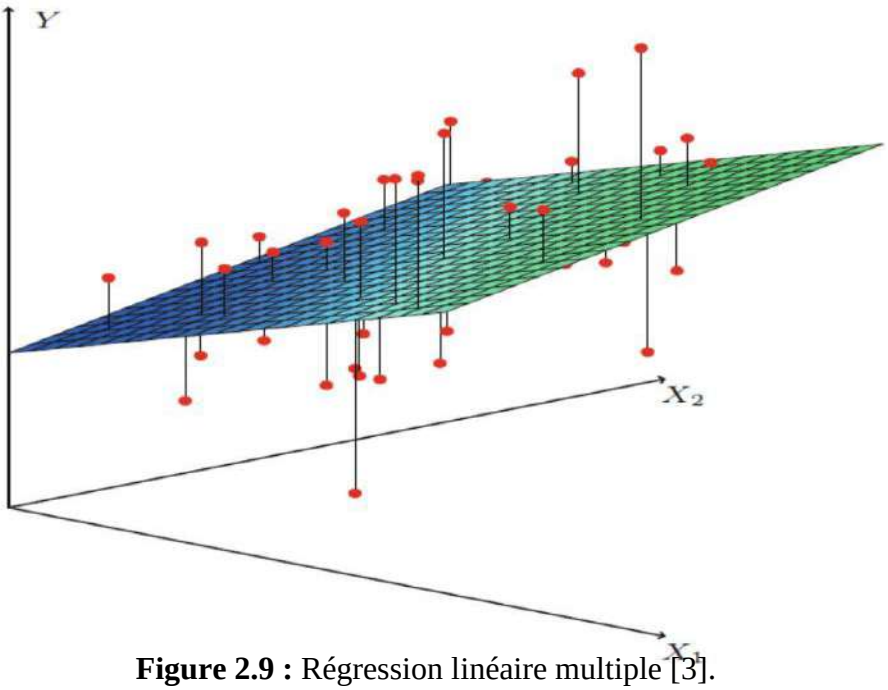


Figure 2.9 : Régression linéaire multiple [3].

Dans un cadre tridimensionnel avec deux prédicteurs et une réponse, la ligne de régression des moindres carrés devient un plan. Choisissez le plan pour minimiser la somme des carrés des distances verticales entre chaque observation (indiquée en rouge) et le plan.

Les valeurs b0'; b1'; :::bp' sont des estimations des coefficients de la régression multivariée des moindres carrés. Contrairement aux estimations de régression linéaire simples données, les estimations des coefficients de régression multiple se présentent sous une forme quelque peu complexe, ce qui est plus facile à exprimer à l'aide de l'algèbre matricielle. [22].

3.4 Machines à vecteurs de support (SVM) :

Une machine à vecteurs de support (SVM) est un algorithme d'apprentissage automatique supervisé principalement utilisé pour la classification. SVM essaie de trouver un hyperplan qui sépare les échantillons dans l'ensemble de données. Sur la figure 2.10, nous pouvons voir deux types de points (rouge et bleu) situés dans un espace d'entités bidimensionnel (axes x et y). Un point est bleu si ses valeurs x et y sont toutes deux inférieures à 5. Dans tous les autres cas, le point est rouge. Dans ce cas, les classes sont séparées linéairement, ce qui signifie que nous pouvons les séparer par un hyperplan.

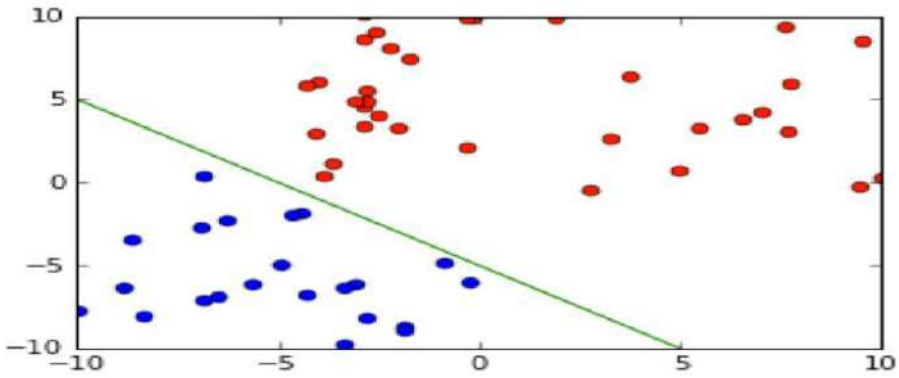


Figure 2.10 : SVM.

SVM essaie de trouver un hyperplan qui maximise la distance entre lui-même et un point. En d'autres termes, parmi tous les hyperplans possibles qui peuvent séparer les échantillons, SVM trouve l'hyperplan avec la plus grande distance de tous les points. .En outre, SVM traite également des données non linéairement séparables. Il y a deux façons : introduire des marges souples ou utiliser des astuces du noyau. [20]
Les marges flexibles fonctionnent en autorisant certains éléments mal classés tout en maintenant la puissance prédictive la plus élevée de l'algorithme. En pratique, il est préférable de ne pas suradapter un modèle d'apprentissage automatique, et nous pouvons le faire en assouplissant certaines hypothèses sur les machines à vecteurs de support.

Les astuces du noyau résolvent le même problème d'une manière différente. Imaginez que nous ayons un espace à deux dimensions, mais les classes sont linéairement inséparables. Les astuces du noyau utilisent les fonctions du noyau pour transformer les données en y ajoutant des dimensions supplémentaires. Dans notre cas, les données transformées seront tridimensionnelles. Une classe qui est linéairement inséparable dans l'espace 2D deviendra linéairement séparable dans l'espace 3D, notre problème est résolu.

3.5 Arbre décision :

Comme les SVM, les arbres de décision sont des algorithmes d'apprentissage automatique à usage général qui peuvent effectuer des tâches de classification et de régression, même des tâches à sorties multiples.

Ce sont des algorithmes très puissants capables de s'adapter à des jeux de données complexes. Les arbres de décision sont également un élément fondamental des forêts aléatoires, l'un des algorithmes d'apprentissage automatique les plus puissants disponibles aujourd'hui [19]. Essentiellement, ils apprennent la hiérarchie des problèmes "if/else" pour prendre des décisions. Ces questions sont similaires à celles qui peuvent être posées dans un jeu de 20 questions. Imaginez que nous voulions différencier les quatre animaux suivants : ours, aigles, pingouins et dauphins. Votre objectif est d'obtenir la bonne réponse avec le moins de questions possible. Nous pouvons commencer par demander si les animaux ont des plumes, une question qui réduit vos chances à seulement deux animaux. Si la réponse est "oui", une autre question peut être posée qui pourrait nous aider à faire la distinction entre les aigles et les pingouins. Par exemple, nous pouvons demander si les animaux peuvent voler. Si les animaux n'ont pas de plumes, vos choix probables sont les dauphins et les ours, et nous devons poser une question pour différencier les deux - par exemple, demander si l'animal a des nageoires.

Cette série de problèmes peut être représentée sous la forme d'un arbre de décision, comme le montre la figure 2.11 ci-dessous.

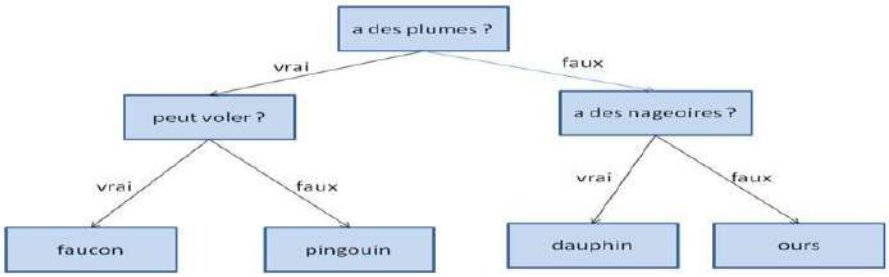


Figure 2.11 : Un arbre de décision pour distinguer entre plusieurs animaux.

Dans ce diagramme, chaque nœud de l'arborescence représente une question ou un nœud terminal (également appelé feuille) qui contient une réponse. Edge relie la réponse à une question à la question suivante que nous allons poser.

Dans Machine Learning Language, nous avons construit un modèle pour utiliser ces trois caractéristiques afin de distinguer quatre classes d'animaux (aigle, pingouin, dauphin et ours)

"Feather", "Flying" et "Finned". Au lieu de construire ces modèles manuellement, nous pouvons les apprendre à partir de données grâce à un apprentissage supervisé.

Les arbres de décision présentent deux avantages par rapport à de nombreux algorithmes : le modèle résultant peut être facilement visualisé et compris par des non-experts (au moins pour les petits arbres), et l'algorithme est complètement invariant à la mise à l'échelle des données. Étant donné que chaque caractéristique est traitée individuellement et que les divisions possibles dans les données sont indépendantes de l'échelle, les algorithmes d'arbre de décision ne nécessitent pas de prétraitement tel que la normalisation ou la normalisation des caractéristiques.
En particulier, les arbres de décision fonctionnent bien lorsque nous avons des caractéristiques d'échelles complètement différentes ou un mélange de caractéristiques binaires et continues.
Le principal inconvénient des arbres de décision est qu'ils ont tendance à sur-ajuster et à fournir de mauvaises performances de généralisation. C'est pourquoi, dans la plupart des applications, une approche d'ensemble comme la forêt aléatoire discutée ci-dessous est généralement utilisée au lieu d'un arbre de décision unique.

3.6 Réseau de neurones artificiels (ANN) :

Les réseaux de neurones artificiels sont des méthodes d'apprentissage supervisées et non supervisées qui tentent d'imiter l'esprit humain grâce à une modélisation simplifiée du système nerveux dans le cerveau humain. Son but n'est pas de simuler le mécanisme exact de la fonction biologique des cellules nerveuses dans le cerveau ou de créer des clones biologiques. Au lieu de cela, la biologie n'est qu'une source d'inspiration. L'élément de traitement unitaire est un modèle simple appelé neurone. Chaque neurone est essentiellement une fonction qui peut recevoir plusieurs entrées et produire une seule sortie. La combinaison de plusieurs neurones dans un réseau de neurones est ce que nous appelons un réseau de neurones artificiels [23].

Ce concept a été introduit pour la première fois en 1943 par McCulloh, qui a proposé la première définition d'un neurone formel [24]. Puis, en 1957, Rosenblatt a créé le premier réseau avec des couches d'entrée et de sortie pour simuler la fonction rétinienne dans la reconnaissance de formes [25]. Depuis lors, avec l'augmentation de la puissance de calcul, un grand nombre de différents types de réseaux de neurones artificiels ont été développés [26]. Quelques exemples d'algorithmes de réseaux de neurones : McCulloch et Pit, les perceptrons, les neurones linéaires adaptatifs (ADALINE), les perceptrons multicouches (MLP), de nombreux neurones linéaires adaptatifs (MADALINE) et les réseaux de rétropropagation.
Les réseaux de neurones artificiels (ANN) peuvent apprendre et donc être formés pour trouver des solutions, reconnaître des modèles, classer des données et prédire des événements futurs. Les réseaux de neurones artificiels sont utilisés pour résoudre des problèmes plus complexes tels que la reconnaissance de caractères, les prévisions boursières et la compression d'images. Le comportement d'un réseau de neurones est défini par la façon dont ses éléments individuels reliés par la force ou le poids de ces connexions. Les pondérations sont automatiquement ajustées en entraînant le réseau selon des règles d'apprentissage spécifiées jusqu'à ce qu'il exécute correctement la tâche souhaitée.
Les réseaux de neurones artificiels sont bien adaptés à la modélisation de données non linéaires avec un grand nombre de caractéristiques d'entrée. Lorsqu'ils sont utilisés correctement, les réseaux de neurones artificiels peuvent résoudre des problèmes difficiles à résoudre avec des algorithmes simples. Cependant, les réseaux de neurones sont coûteux en calcul et il est difficile de comprendre comment les réseaux de neurones artificiels parviennent à des solutions.

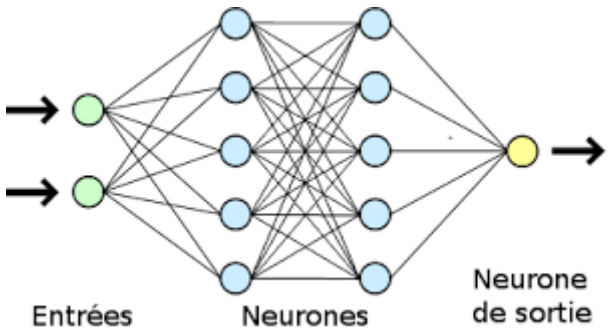


Figure 2.12 : Réseau de neurones artificiels (ANN).

3.7 Les arbres aléatoires :

Comme nous venons de l'observer, l'un des principaux inconvénients des arbres de décision est qu'ils ont tendance à surajuster les données d'apprentissage. Les forêts aléatoires sont un moyen de résoudre ce problème. Une forêt aléatoire est essentiellement une collection d'arbres de décision, où chaque arbre est légèrement différent des autres. L'idée derrière les forêts aléatoires est que chaque arbre peut bien prédire, mais peut surapprendre certaines données. Si nous construisons plusieurs arbres, qui fonctionnent tous bien et se chevauchent de différentes manières, nous pouvons réduire le nombre de super additions en faisant la moyenne de leurs résultats. Cette réduction du surajustement, tout en préservant le pouvoir prédictif de l'arbre, peut être démontrée par des calculs mathématiques rigoureux.
Pour mettre en œuvre cette stratégie, nous devons construire de nombreux arbres de décision.

Chaque arbre doit prédire la cible d'une manière acceptable et doit également être différent des autres arbres. Les forêts aléatoires tirent leur nom de l'injection de caractère aléatoire dans la construction des arbres pour s'assurer que chaque arbre est différent. Les arbres dans les forêts aléatoires sont randomisés de deux manières : en choisissant les points de données utilisés pour construire l'arbre et en choisissant les caractéristiques pour chaque test fractionné.
Random Forest est l'algorithme le plus connu de la technique d'apprentissage d'ensemble "Bagging" ou "Guided Aggregation", qui consiste à créer plusieurs entités à partir d'un même modèle (plusieurs arbres de décision) et à entraîner chacun de ces arbres. Collecte des données, après avoir entraîné chaque arbre, on peut regrouper les résultats de chaque arbre afin de faire des prédictions, la figure 1.10 montre la technique du "bagging".

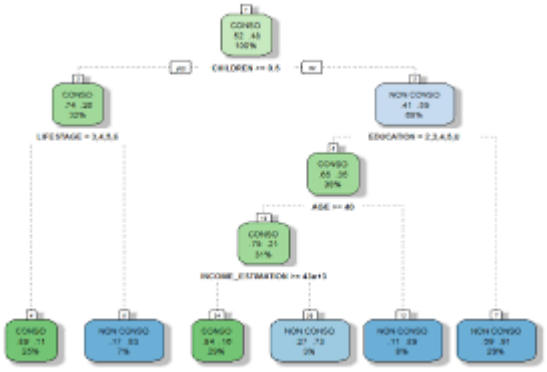


Figure 2.13 : Les arbres aléatoires

4. Conclusion

Dans ce chapitre, nous avons présenté les grands axes de l'apprentissage automatique de manière globale, quelques définitions, ses types, et quelques algorithmes spécifiques à l'apprentissage supervisé.

Chapitre 03 : les techniques de classification en détection d’intrusion

Introduction :

Après avoir présenté dans les chapitres précédents l'état de l'art d'une part sur le domaine d'intérêt à savoir la sécurité informatique et la détection d'intrusion, et d'autre part le paradigme de calcul, à savoir les méthodes d'apprentissage automatiques, nous présentons dans ce chapitre une études conceptuelle concernant les techniques de classification en détection d'intrusion, et dont le but est de comparer classifieurs afin de déterminer lequel est plus adapté à la détection d'intrusion, en utilisant la célèbre base de données KDD.

1. Problématique :

Notre problématique consiste donc à l'étude de quelques classifieurs supervisés pour la détection avec utilisation de data set KDDCUP99. Le choix de l'approche supervisée est motivé par le fait que les donnes d'intrusion de la base de données KDD sont étiquetées. C'est-à-dire pour chaque entrée de la base, l'attaque est bien spécifiée sous forme d'une classe (DOS, ICMP, … etc.).

2. Méthode d'évaluation :

Plusieurs métriques existent pour l'évaluation d'une classification de données en se basant sur les données de réalité terrain dans un contexte de classification supervisée. Ces métriques se basent sur quelques métriques de base et qui sont : le nombre d'instances correctement classées positivement (TP : True Positives), le nombre d'instances incorrectement classées positivement (FP : false positives), le nombre d'instances incorrectement classées négatives (FN : false négatives). Plusieurs métriques de synthèse ont été définies en se basant sur ces métriques de base, telle que la métrique de Jaccard et la métrique de Dice (Kappa index), que nous définissions en détail dans la suite de la section.

2.1. Métrique de Dice :

L'indice est connu sous divers autres noms : le plus souvent reviennent indice de Sorensen ou coefficient de Dice ; les deux noms se voient aussi avec le qualificatif « coefficient de similarité » ou « indice » ou autres variations, et le nom « Sorensen » est orthographié avec diverses variations, comme « Sorenson », « Soerenson » ou «

Sorensen », et dans chaque cas le suffixe peut être remplacé par « sen ». On trouve également le nom indice binaire Czekanowski.
L'indice mesure la présence ou l'absence d'espèces. On peut étendre l'expression à la mesure de l'abondance au sens écologique du terme. Des versions quantitatives sont connues sous divers noms :

- Indice quantitatif de Sorensen-Dice, de Sorensen, de Dice.
- Distance de Bray-Curtis (l'opposée de la dissimilarité de Bray-Curtis).
- Indice quantitatif de Czekanowski, de Steinhaus.
- Similarité en pourcentage de Pielou (en).
- L'opposée de la distance de Hellinger **[27]**.

3.Le Data set KDDcup99 :

Avec l'énorme croissance de l'utilisation des réseaux informatiques et l'énorme augmentation du nombre d'applications exécutées dessus, la sécurité du réseau devient de plus en plus importante. tous les systèmes informatiques souffrent de vulnérabilités de sécurité qui sont à la fois techniquement difficiles et économiquement coûteuses à résoudre par les fabricants. Par conséquent, le rôle des systèmes de détection d'intrusion (IDS), en tant que dispositifs spéciaux pour détecter les anomalies et les attaques sur le réseau, devient de plus en plus important. La recherche dans le domaine de la détection d'intrusions s'est principalement concentrée sur les techniques de détection basées sur les anomalies et les abus pendant longtemps. Alors que la détection basée sur une mauvaise utilisation est généralement favorisée dans les produits commerciaux en raison de sa prévisibilité et de sa grande précision, dans la recherche universitaire, la détection des anomalies est généralement conçue comme une méthode plus puissante en raison de son potentiel théorique pour traiter de nouvelles attaques **[28].**

3.1 Description des Donnes et Prétraitement :

Depuis 1999, KDDCup 99 est utilisé comme ensemble de données exemple dans les systèmes de détection d'intrusion comportementale **[4].** Chaque paquet de l'ensemble des données de KDD Cup 99 est constitué de 41 champs et est labellisés comme paquet normal ou paquet anormal avec les types d'attaques. Parmi ces champs, 37 sont des champs de type

numérique et 4 sont des champs de type non numérique. KDD99 regroupe 37 types d'attaque. Ces attaques sont divisées en quatre grandes classes : DOS, U2R, R2L et Probes.

- **DOS (Denial of service attaks) :** ce sont les attaques qui visent à porter atteinte à la disponibilité des services en saturant les ressources de la machine, serveur ou réseau cibles. Ces attaques réussies dans les réseaux ont pour conséquence immédiate le blocage du trafic réseaux.
- **Probes :** attaque qui vise à réunir les informations sur la cible susceptible d'aider l'attaquant à initier une attaque. Il existe plusieurs types d'attaques probes : certains abusent les utilisateurs légitime et d'autres utilisent la technique d'ingénierie pour collecter les informations.
- **R2L (Remote To Local) :** attaque qui vise à contourner ou usurper les paramètres d'authentification d'une cible afin d'exécuter les commandes. La plupart de ces attaques sont issues de la sociale ingénierie **[29]**.
- **U2R (User To Root) :** l'attaque provient ici de l'intérieur. L'attaquant usurpe le mot de passe du super administrateur par conséquent des autres utilisateurs. La plupart de ces attaques sont issues de la saturation du Buffer causée par les erreurs de programmation **[29]**.

Il est important de noter que les données de test ne proviennent pas de la même distribution de probabilité que les données d'apprentissage, et qu'elles incluent des types d'attaques spécifiques ne figurant pas dans les données d'apprentissage, ce qui rend la tâche plus réaliste. Certains experts en intrusion pensent que la plupart des nouvelles attaques sont des variantes d'attaques connues et que la signature d'attaques connues peut être suffisante pour capturer de nouvelles variantes. Les ensembles de données contiennent un nombre total de 24 types d'attaques d'entraînement, avec 14 types supplémentaires dans les données de test uniquement **[28].**

3.2 Problèmes Inhérents Du Jeu De Données Kdd'99 :

Comme mentionné dans la section précédente, KDD'99 est construit sur la base des données capturées dans DARPA'98 qui a été critiquée par McHugh **[4],** principalement en raison des caractéristiques des données synthétiques. En conséquence, certains des problèmes existants dans DARPA'98 demeurent dans KDD'99. Cependant, il y a des améliorations délibérées ou non intentionnelles, ainsi que des problèmes supplémentaires. Dans ce qui suit, nous passons d'abord en revue les problèmes dans DARPA'98, puis nous

discutons de l'existence possible de ces problèmes dans KDD'99. Enfin, nous discutons des nouveaux problèmes observés dans l'ensemble de données KDD.

- Dans un souci de confidentialité, les expériences ont choisi de synthétiser à la fois les données de fond et les données d'attaque, et les données seraient similaires à celles observées pendant plusieurs mois d'échantillonnage de données à partir d'un certain nombre de bases de l'armée de l'air. Cependant, aucune validation analytique ou expérimentale des caractéristiques de fausse alarme des données n'a été entreprise. De plus, la charge de travail des données synthétisées ne semble pas similaire au trafic dans les réseaux réels.
- Les collecteurs de trafic tels que TCPdump, qui est utilisé dans DARPA'98, sont très susceptibles de devenir surchargés et de rejeter des paquets avec une charge de trafic importante. Cependant, il avait aucun examen pour vérifier la possibilité des paquets perdus.

Il n'y a pas de définition exacte des attaques. Par exemple, le sondage n'est pas nécessairement un type d'attaque à moins que le nombre d'itérations dépasse un seuil spécifique. De même, un Paquet qui provoque un débordement de tampon n'est pas toujours représentatif d'une attaque. Dans ces conditions, il devrait y avoir un accord sur les dé finitions entre l'évaluateur et l'évalué. Dans DARPA'98, cependant, il n'existe pas de dé finitions spécifiques des attaques réseau.

4. Classifieur à tester :

4.1 Le classifieur Naïve Bayes :

- **Algorithme :**

Etant donné une base de connexion C = {c1, c2…}, la tâche d'apprentissage est de décider pour chaque connexion si elle suspecte ou non. La décision est basée sur le mécanisme d'inférence d'un modèle probabiliste M, ou un sous-ensemble de propriétés des connexion et donné comme entrée de ce modèle, telles que : adresse, protocole, service, durée, temps, etc.

Dans notre contexte, pour classer une nouvelle connexion, appliquer le classifieur bayésien, qui va calculer la probabilité liée à cette connexion. Puis, selon un seuil donné, nous décidons si cette connexion est suspecte ou non.

Ou c'est une valeur possible de la classe C et A est l'information sur les attributs.

Si P(ci\A) est la sortie du classifieur de Bayes et s est le seuil (par exemple s= 50%), le pseudo code suivant (Algorithme illustre la méthode de classification d'une connexion ci :

Entrée : Connexion ci, avec un ensemble de propriétés A ; le seuil s

Sortie : Classe de connexion.

Calculer P (ci\A) = $(A/B) = \frac{P(B/A) .P(A)}{P(B)}$

Si (P (ci\A) > s) **alors**

Classe= vrai ; connexion suspecte

Sinon

Classe= faux ; connexion non suspecte

Fin si

4.2 Le classifieur arbre décision C4.5 :

- **Algorithme :**

Avec en entrée, un échantillon de m enregistrements classés (x ;c(X), un algorithme d'apprentissage doit fournir en sortie un arbre de décision.

Donnée : Un échantillon S de m enregistrement classés (x ;c(X)).

Résultat : Arbre de décision élagué.

Début

- Initialisation :

Arbre ← vide

Nœud courant ← racine

Echantillon courant ← S

Répéter :

- Décider si le nœud courant est terminal :

Si le nœud courant est terminal **alors**

Etiqueter le nœud courant par une feuille.

Sinon

Sélectionner un test et créer le sous arbre.

Nœud courant ← un nœud non encore étudié.

Echantillon courant ← échantillon atteignant le nœud courant.

Jusqu'à production d'un arbre de décision.

Elaguer l'arbre de décision obtenu.

Fin

4.3 Le classifieur arbre aléatoire :

- **Algorithme :**

L'algorithme des « arbre aléatoires » (ou Random tree) est un algorithme de classification qui réduit la variance des prévisions d'un arbre de décision seul, améliorant ainsi leurs performances. Pour cela, il combine de nombreux arbres de décisions dans une approche de type bagging.

L'algorithme des « arbre aléatoires » a été proposé par Leo Breiman et Adèle Cutler en 2001. Dans sa formule la plus classique, il effectue un apprentissage en parallèle sur de multiples arbres de décision construits aléatoirement et entraînés sur des sous-ensembles de données différents. Le nombre idéal d'arbres, qui peut aller jusqu'à plusieurs centaines voire plus, est un paramètre important : il est très variable et dépend du problème. Concrètement, chaque arbre de la forêt aléatoire est entrainé sur un sous ensemble aléatoire de données selon le principe du bagging, avec un sous ensemble aléatoire de features (caractéristiques variables des données) selon le principe des « projections aléatoires ». Les prédictions sont ensuite moyennées lorsque les données sont quantitatives ou utilisés pour un vote pour des données qualitatives, dans le cas des arbres de classification. L'algorithme des forêts aléatoires est connu pour être un des classifieurs les plus efficaces « out-of-the-box » (c'est-à-dire nécessitant peu de prétraitement des données). Il a été utilisé dans de nombreuses applications, y compris grand public, comme pour la classification d'images de la caméra de console de jeu Kinect* dans le but d'identifier des positions du corps.

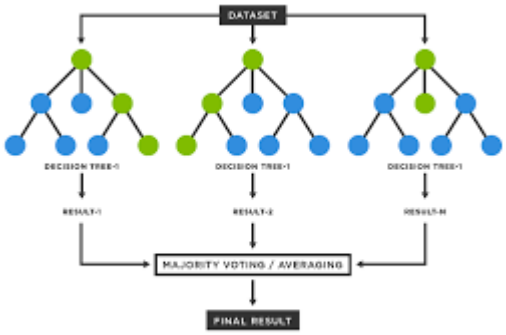


Figure3.1 : Exemple de classification arbre aléatoire**.**

4.4 Le classifieur K-plus proche voisin (K-NN) :

- **Algorithme :**

Les exemples d'apprentissage sont des vecteurs dans un espace de caractéristiques multidimensionnel, chacun avec une étiquette de classe d'appartenance. La phase d'apprentissage de l'algorithme consiste seulement dans le stockage des vecteurs caractéristiques et des étiquettes de classe des échantillons d'apprentissage.

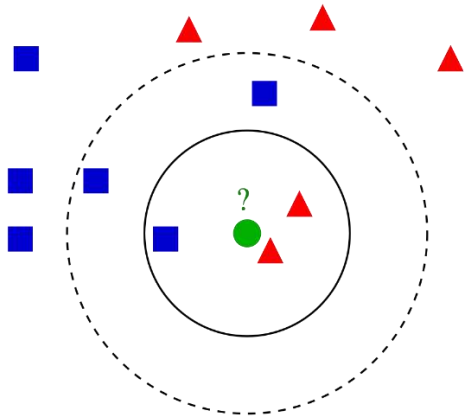


Figure3.2 : Exemple de classification K-nn.

Dans la phase de classification, *k* est une constante définie par l'utilisateur, et un vecteur non étiqueté (une requête ou un point de test) est classé en lui affectant l'étiquette qui est la plus fréquente parmi les *k* échantillons d'entraînement les plus proches du point à classer.

La distance commune pour des variables continues est la distance euclidienne. Pour des variables discrètes, comme en classification de texte, une autre distance peut être utilisée, telle que la distance de recouvrement (ou la distance de Hemming). Dans le contexte de micro-tableau de données génétiques, par exemple, K-nn a aussi été employée avec des coefficients de corrélation de Pearson et Spearman **[30]**.

Fréquemment, la précision de la classification K-nn peut être améliorée de manière significative si la distance est apprise par des algorithmes spécialisés tels que la méthode du plus proche voisin à grande tolérance ou l'analyse des composantes de voisinage.

Une faiblesse de la classification dans sa version de base par vote majoritaire apparaît quand la distribution de classe est asymétrique. C'est-à-dire, des exemples d'une classe plus fréquente tendent à dominer la prédiction de classification du nouvel entrant, car elle serait statistiquement plus fréquente parmi les k plus proches voisins par définition **[31]**.

Un moyen de surmonter cette difficulté est de pondérer la classification, en prenant en compte la distance du nouvel entrant à chacun de ses k plus proches voisins. La classe (ou la valeur en cas de régression) de chacun de ces k plus proches voisins est multipliée par une pondération proportionnelle à l'inverse de la distance de ce voisin au point à classer. Une autre façon de s'affranchir de cette asymétrie se fait par abstraction dans la représentation des données. Par exemple, dans une carte auto adaptative (SOM), chaque nœud est représentatif du centre de gravité (barycentre) d'un amas de points similaires, indépendamment de leur densité dans les données originales d'apprentissage. La méthode K-nn peut être employée pour les SOM.

5.clustering a tester

5.1 Le cluster K-means:

- **Algorithme :**

Entrée :

- K le nombre de cluster à former
- Le Training Set (matrice de données)

DEBUT

Choisir aléatoirement K points (une ligne de la matrice de données). Ces points sont les centres des clusters (nommé centroïd).

REPETER

Affecter chaque point (élément de la matrice de donnée) au groupe dont il est le plus proche au son centre

Recalculer le centre de chaque cluster et modifier le centroide

JUSQU'A convergence

OU (stabilisation de l'**inertie totale** de la population)

FIN ALGORITHME

Note 1 : Lors de la définition de l'algorithme, quand je parle de "point", c'est un point au sens "donnée/data" qui se trouve dans un espace vectoriel de dimension . Avec : le nombre de colonnes de la matrice de données.

Note 2 : La convergence de l'algorithme K-Means peut être l'une des conditions suivantes :

- Un nombre d'itérations fixé à l'avance, dans ce cas, K-means effectuera les itérations et s'arrêtera peu importe la forme de clusters composés.
- Stabilisation des centres de clusters (les centroïdes ne bougent plus lors des itérations).

L'affectation d'un point à un cluster se fait en fonction de la distance de ce point par rapport aux différents centroïdes. Par ailleurs, ce point se fera affecter à un cluster s'il est plus proche de son centroïde (distance minimale). Finalement, la distance entre deux points dans le cas de K-Means se calcule par les méthodes évoquées dans le paragraphe "notion de similarité".

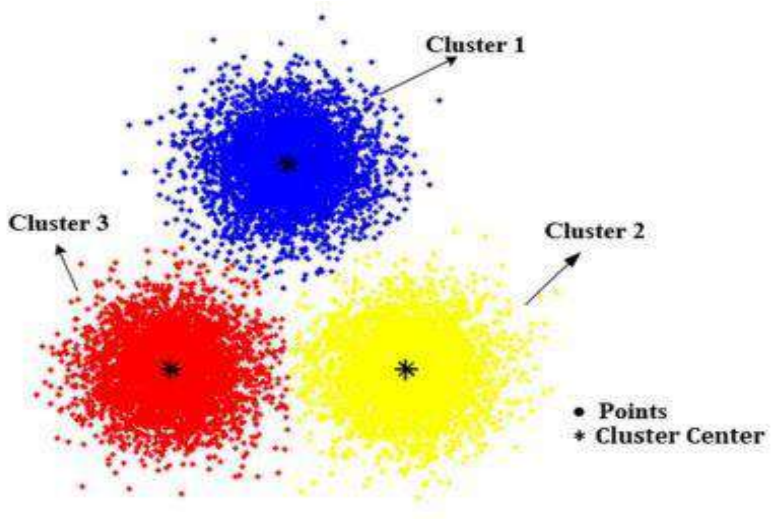


Figure 3.3 : exemple de classification on k-means.

6.Le Protocole expérimental :

Comme nous l'avons indiqué plus haut, nous expérimentons deux algorithmes de classification supervisée pour les données de détection d'intrusion, issues de la base de données spécialisée, KDD dans sa version KDDCup99. Cependant, et à cause de la taille très élevée de cette base de données, nous avons opté pour une version light de cette base, et qui consiste à 20% de la base initiale. Le raffinement des données de la base KDD, consiste à éliminer les données qui sortent d'un domaine d'intérêt et de sélectionner un sous ensemble de données pour l'apprentissage.

Figure 3.4 : Diagramme du protocole expérimental.

7.Partition de la base KDD cup99 :

La version 20% de la KDD a été considéré dans notre travail de comparaison. Nous l'avons partionné en un sous ensemble d'apprentissage et un sous ensemble de test selon les pourcentages suivants :

- 66.66% pour l'apprentissage.
- 33.33% pour le test.

Cependant, et pour ne bas biaiser les données d'apprentissage et de test, nous procédons par un échantillonnage des données d'apprentissage d'une manière aléatoire, comme l'indique l'algorithme suivant :

Algorithme 1 : Echantillonnage – Apprentissage.

Début

Taille_ apprentissage ← 2*KDD/3

Pour i 1 ← àTaille_ apprentissage **faire**

J ← aléatoire (Taille KDD)

Ensemble d'apprentissage[i] ← KDD[j]

Fin Pour

Fin

Algorithme 2 : Echantillonnage – Test.

Début

Taille_ Test ← Taille (KDD)/3

Pour i← 1 àTaille_ Test **faire**

J ← aléatoire (Taille KDD)

Ensemble Test [i] ← KDD [j]

Fin Pour

Fin

8. Diagramme de classe :

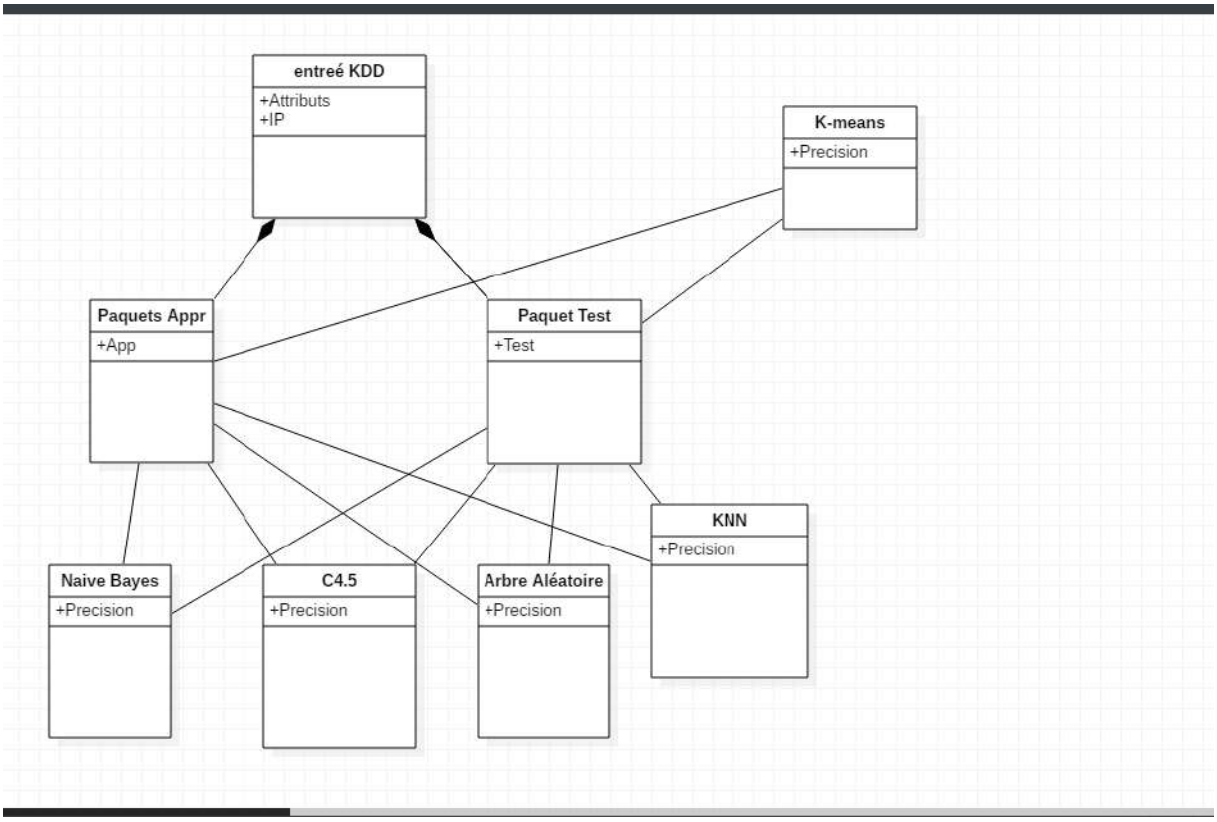


Figure 3.5 : Diagramme de classe

9.Conclusion :

Dans ce chapitre, nous avons présenté une démarche expérimentale pour tester quelques classifieurs avec la base de données de sécurité KDD. Nous avons considéré quatre classifieurs, à savoir le classifieur **naive bayes**, et le classifieur **C 4.5**, et le classifieur **arbre aléatoire**, et le classifieur **KNN** et le cluster **K-means**. Dans une étude plus détaillée, il serait préférable de tester plus de classifieurs afin de décider quel est le classifieur le plus approprié pour les données de la KDD.

Au chapitre suivant nous montrons les aspects pratique de l'expérimentation, à savoir la présentation de la plateforme de test Weka et java sous NetBeans, et nous présentons également quelques résultats de test.

Chapitre 04 : implémentation et test

Introduction

Dans ce chapitre, nous allons présenter nos résultats expérimentaux afin étudier la performance de l'apprentissage dans la détection d'intrusion.

Nous allons décrire les expérimentations avec l'ensemble d'apprentissage KDDCup99, et nous terminerons par une étude comparative entre les différents classifieurs utilisés. Dans ce qui suit, nous allons présenter les différents outils logiciels, qui nous ont permis d'implémenter notre modèle de détection d'intrusion.

1.La plateforme Weka :

Weka (Waikato Environnent for Knowledge Analysis) est un ensemble d'outils permettant de manipuler et d'analyser des fichiers de données implémentant la plupart des algorithmes d'intelligence artificielle, entre autres, les arbres de décision et les réseaux de neurones.

Il est écrit en java, disponible sur le web, et s'appuie sur le livre :

/« Data Mining, practical machine learning tools and techniques with Java implementations Witten & Frank, Editeur: Morgan Kauffman ».

Il se compose principalement :

- De classes Java permettant de charger et de manipuler les données.
- De classes pour les principaux algorithmes de classification supervisée ou non supervisée.
- D'outils de sélection d'attributs, de statistiques sur ces attributs.
- Via l'interface graphique, pour charger un fichier de données, lui appliquer un algorithme, vérifier son efficacité.
- Invoquer un algorithme sur la ligne de commande.
- Utiliser les classes définies dans ses propres programmes pour créer d'autres méthodes, implémenter D'autres algorithmes, comparer ou combiner plusieurs méthodes [**26]**.

1.1. Description :

L'espace de travail Weka **[32]** contient une collection d'outils de visualisation et d'algorithmes pour l'analyse des données et la modélisation prédictive, allié à une interface graphique pour un accès facile de ses fonctionnalités. La version « non-Java » originale de Weka était un front-end en Tcl/Tk pour des algorithmes de modélisation (essentiellement tierces) implémentés dans d'autres langages de programmation, complété par un des utilitaires de préprocesseur de données en C, et un système à base de makefile pour lancer les expériences d'apprentissage automatique. Cette version originale était avant tout conçue comme un outil pour analyser des données agricoles**[33]**,**[34]**, mais la version plus récente entièrement basée sur Java (Weka 3), pour laquelle le développement a débuté en 1997, est désormais utilisée dans beaucoup de domaines d'application différents, en particulier pour l'éducation et la recherche. Les principaux points forts de Weka sont :

- est libre et gratuit, distribué selon les termes de la licence publique générale GNU .
- est portable car il est entièrement implémenté en Java et donc fonctionne sur quasiment toutes les plateformes modernes, et en particulier sur quasiment tous les systèmes d'exploitation actuels .
- Contient une collection complète de préprocesseurs de données et de techniques de modélisation.
- Est facile à utiliser par un novice en raison de l'interface graphique qu'il contient.

Weka supporte plusieurs outils d'exploration de données standards, et en particulier, des préprocesseurs de données, des agrégateurs de données (data clustering), des classificateurs statistiques, des analyseurs de régression, des outils de visualisation, et des outils d'analyse discriminante. Toutes les techniques de Weka reposent sur la supposition que les données sont disponibles dans un unique fichier plat ou une Relation binaire, ou chaque type de donnée est décrit par un nombre fixe d'attributs (les attributs ordinaires, numériques ou symboliques, mais quelques autres types d'attributs sont aussi supportés). Weka fournit un accès aux bases de données SQL en utilisant le Java Database Connectivity (JDBC) et peut traiter le résultat d'une requête SQL. Il n'est pas capable de faire de l'exploration de données multi-relationnelles, mais il existe des logiciels tiers pour convertir une collection de tables de base

de données liées entre elles en une table unique adaptée au traitement par Weka **[35]**. Un autre domaine important qui n'est pour le moment pas couvert par les algorithmes inclus dans la distribution Weka est la modélisation de séquences.

L'interface principale de Weka est l'explorer, mais à peu près les mêmes fonctionnalités peuvent être atteintes via l'interface « flux de connaissance » de chaque composant et depuis la ligne de commande. Il y a aussi l'experimenteur, qui permet la comparaison systématique (taxinomique) des performances prédictives des algorithmes d'apprentissage automatique de Weka sur une collection de jeux de données.

L'interface explorer possède plusieurs onglets qui donnent accès aux principaux composants de l'espace de travail. L'onglet préprocesseur a plusieurs fonctionnalités d'import de données depuis des bases de données, un fichier CSV et pour prétraiter ces données avec un algorithme appelé filtering. Ces filtres peuvent être utilisés pour transformer les données (par exemple, transformer des attributs numériques réels en attributs discrets) et rendre possible l'effacement d'instances et d'attributs selon des critères spécifiques. L'onglet classify permet à l'utilisateur d'appliquer des classifications et des algorithmes de régression (indifféremment appelés « classifiers » dans Weka) au jeu de données résultant, pour estimer la précision du modèle prédictif, et de visualiser les prédictions erronées, ROC curves, etc. ou le modèle lui-même (si le modèle est sujet à visualisation, comme un Arbre de décision). L'onglet Associate donne accès aux apprentissages par règles d'association qui essayent d'identifier toutes les relations importantes entre les attributs dans les données. L'onglet Cluster donne accès aux techniques de clustering de Weka, comme l'algorithme du k-means. Il y a aussi une implémentation d'algorithme espérance-maximisation pour l'apprentissage d'un mélange de distributions normales. L'onglet « Select attributes » fournit des algorithmes pour l'identification des attributs les plus prédictifs dans un jeu de données. Le dernier onglet, « Visualize » montre une matrice de nuages de points, ou des nuages de points individuels peuvent être sélectionnés et élargis, et davantage analysés en utilisant divers opérateurs de sélection.

Figure4.1 : L'interface graphique du logiciel Weka.

2. Netbeans :

Pour l'environnement de développement de notre application, nous avons eu recours au logiciel Netbeans.

Netbeans est un environnement de développement un outil pour les programmeurs pour écrire, compiler, déboguer et déployer des programmes. Il est écrit en Java - mais peut supporter n'importe quel langage de programmation. Il y a également un grand nombre de modules pour étendre l'EDI Netbeans. L'EDI Netbeans est un produit gratuit, sans aucune restriction quant à son usage.

Egalement disponible, La Plateforme Netbeans; une fondation modulable et extensible utilisée comme brique logicielle pour la création d'applications bureautiques. Les partenaires privilégiés fournissent des modules à valeurs rajoutées qui s'intègrent facilement à la Plateforme et peuvent être utilisés pour développer ses propres outils et solutions.

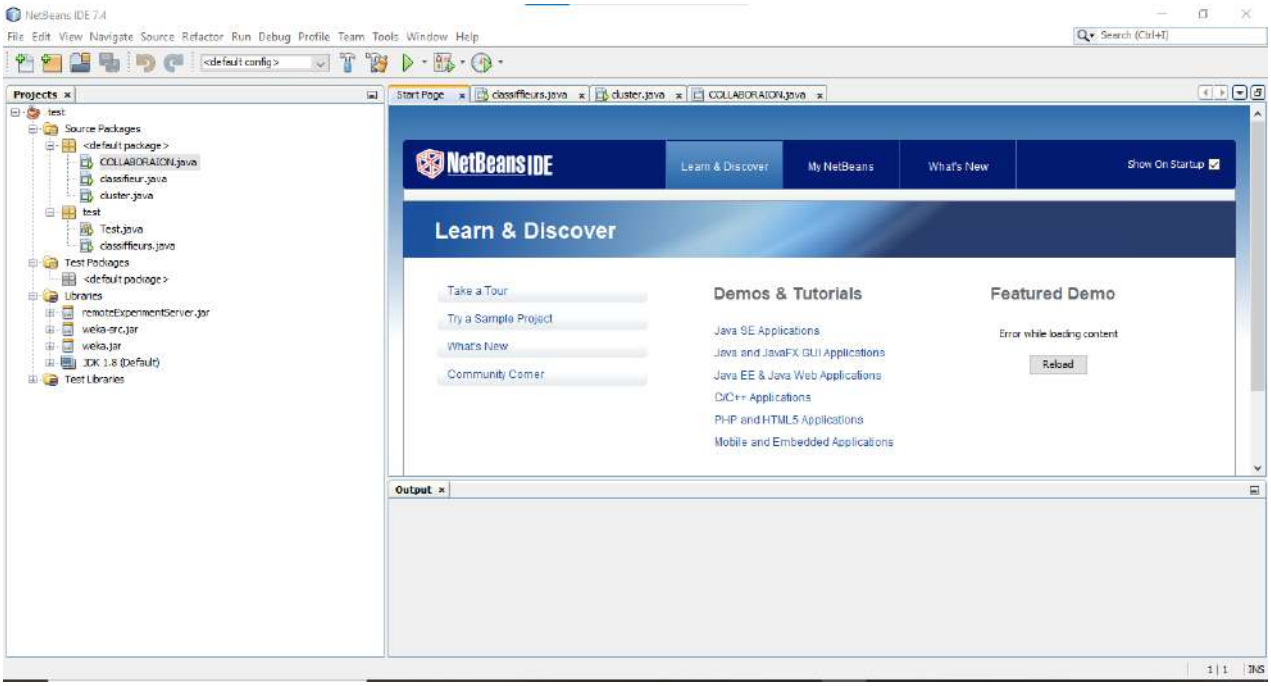


Figure4.2 : Interface NetBeans

3. Expérimentation :

3.1. Chargement de KDD :

Comme nous l'avons mentionné au chapitre 3, nous avons travaillé avec la version 20% de la base de données d'intrusion KDD. La figure 4.3 montre le chargement du fichier correspondant.

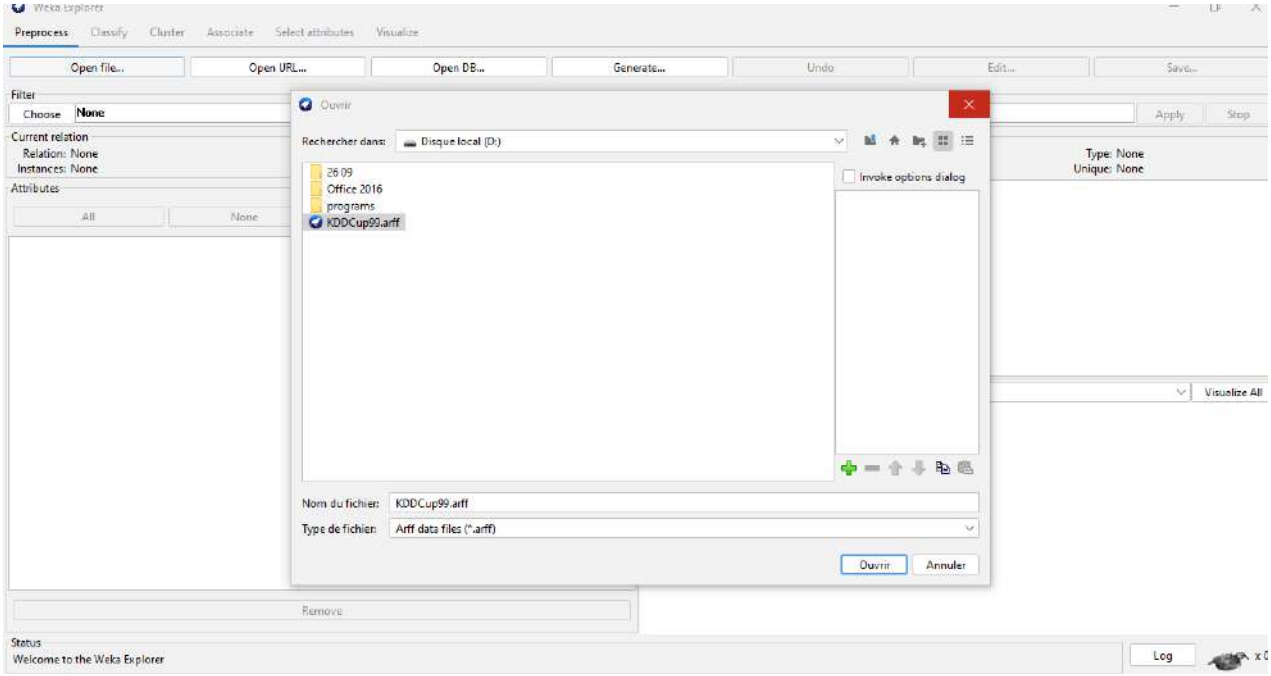


Figure4.3 : Chargement de la base KDD.

3.2. Section attributs :

Les données d'intrusion sont décrites par plusieurs attributs relatifs au trafic réseau, capturé sur des ports d'entrée. Il est rare que tous les attributs soient pris en compte lors de la classification, et on se contente donc à quelques attributs dont on juge qu'ils sont pertinents pour la détection d'intrusions. Dans notre cas, nous avons pris en compte tous les attributs de la base KDD.

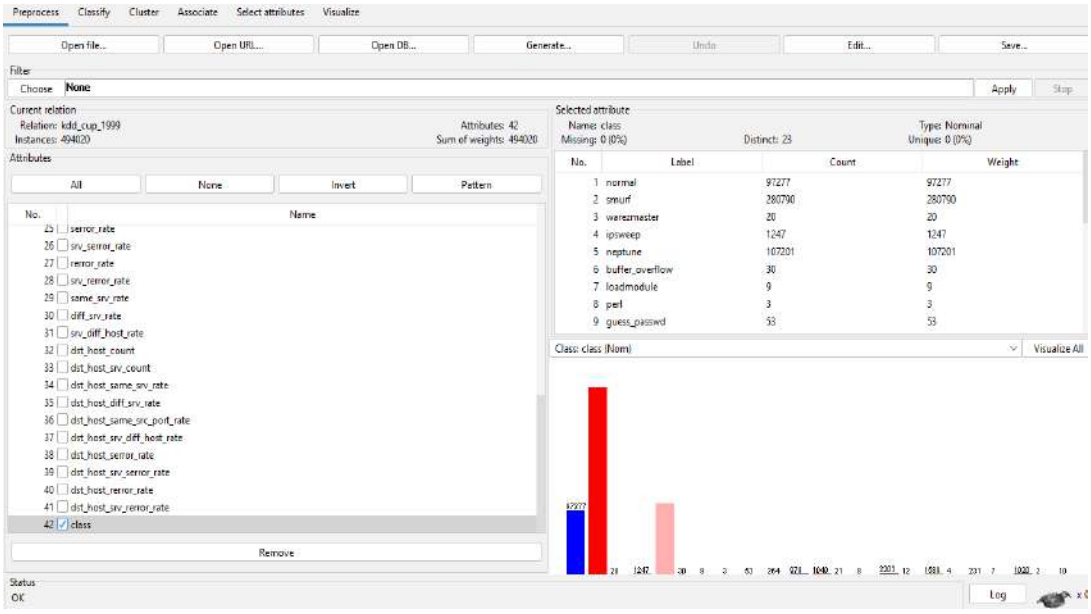


Figure 4.4: Sélection d'attributs.

3.3. Test avec le classifieur naïve bayes :

La figure suivante montre la sélection du classifieur **naïve bayes** dans Weka.

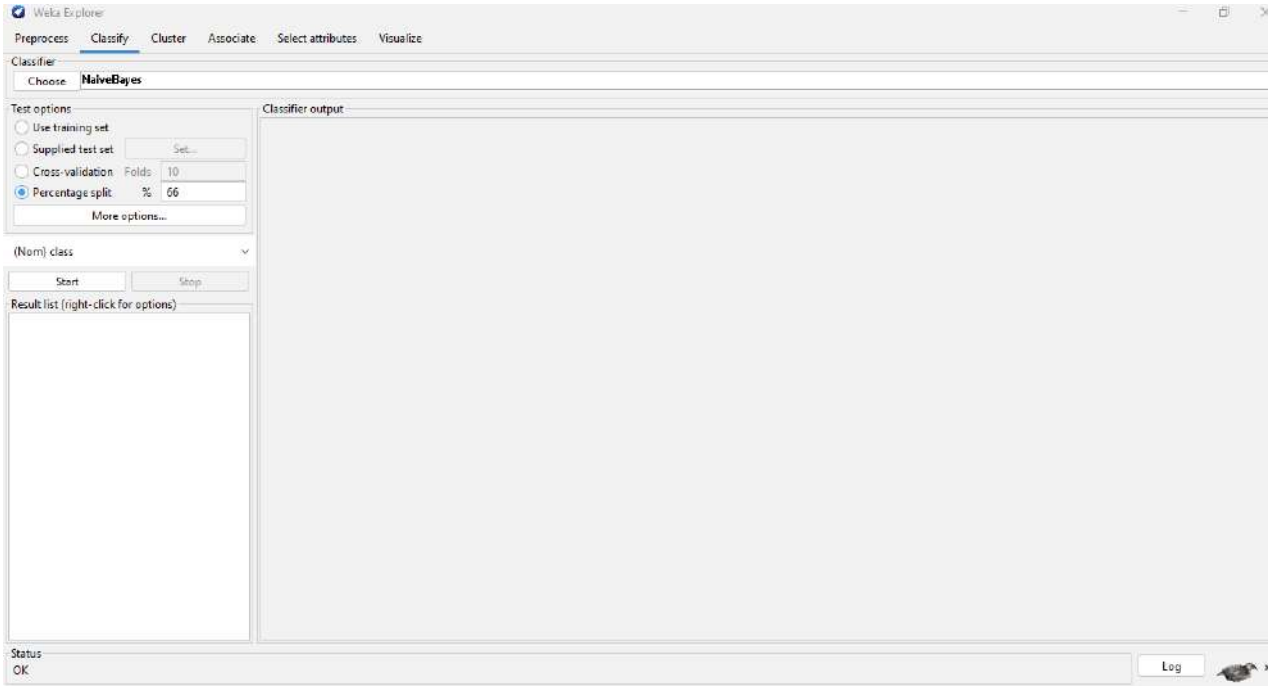


Figure 4.5: Sélection du classifieur naïve bayes.

3.3.1. Résultats de test naïve bayes :

La figure 4.6 montre l'interface d'affichages des résultats de classification de données KDD.

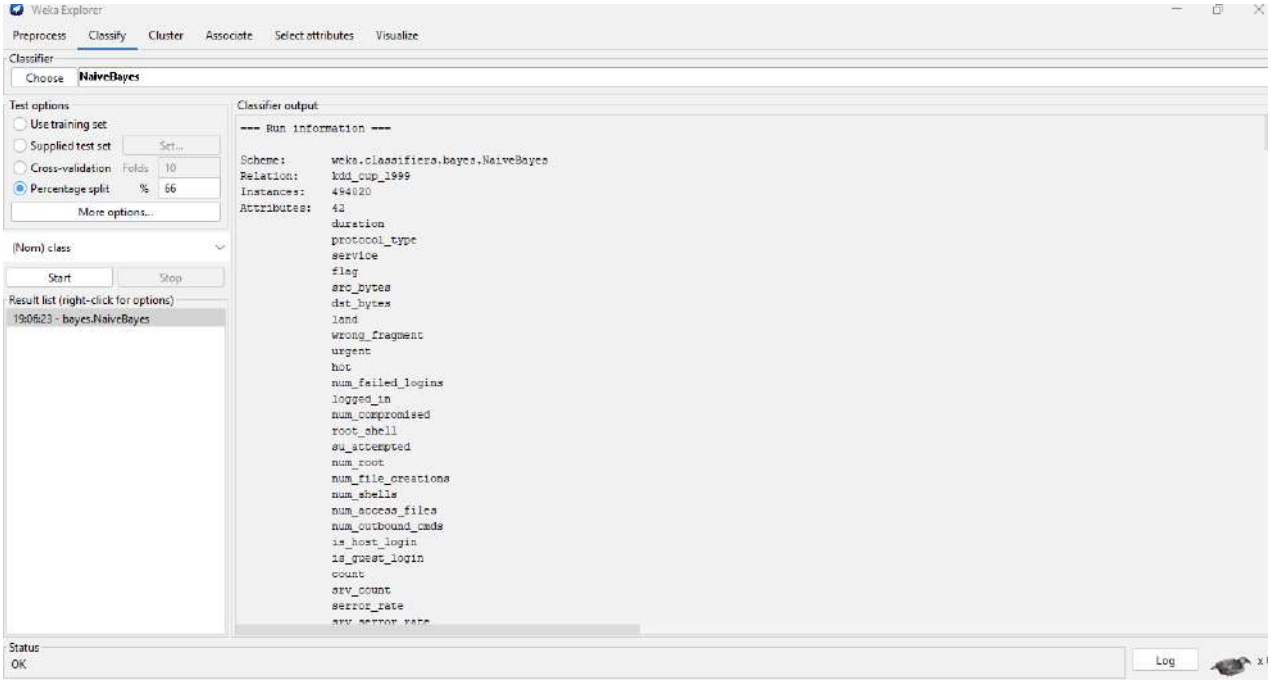


Figure 4.6: Résultats avec le classifieur **naïve** bayes.

Nous avons développé également une interface propre à nous qui fait appel aux bibliothèques de Weka. Elle a été implémenté en utilisant le langage Java sous Netbeans.

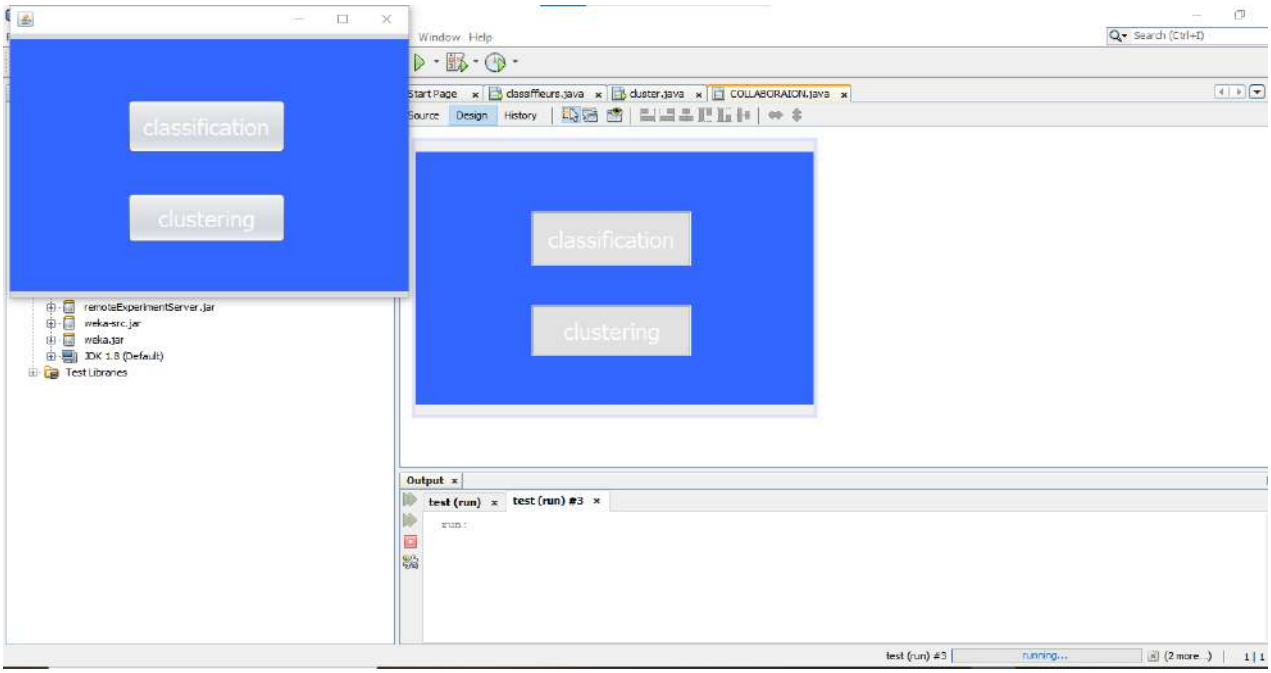


Figure 4.7 : interface d'accueil

- **Click sur le Button classification**

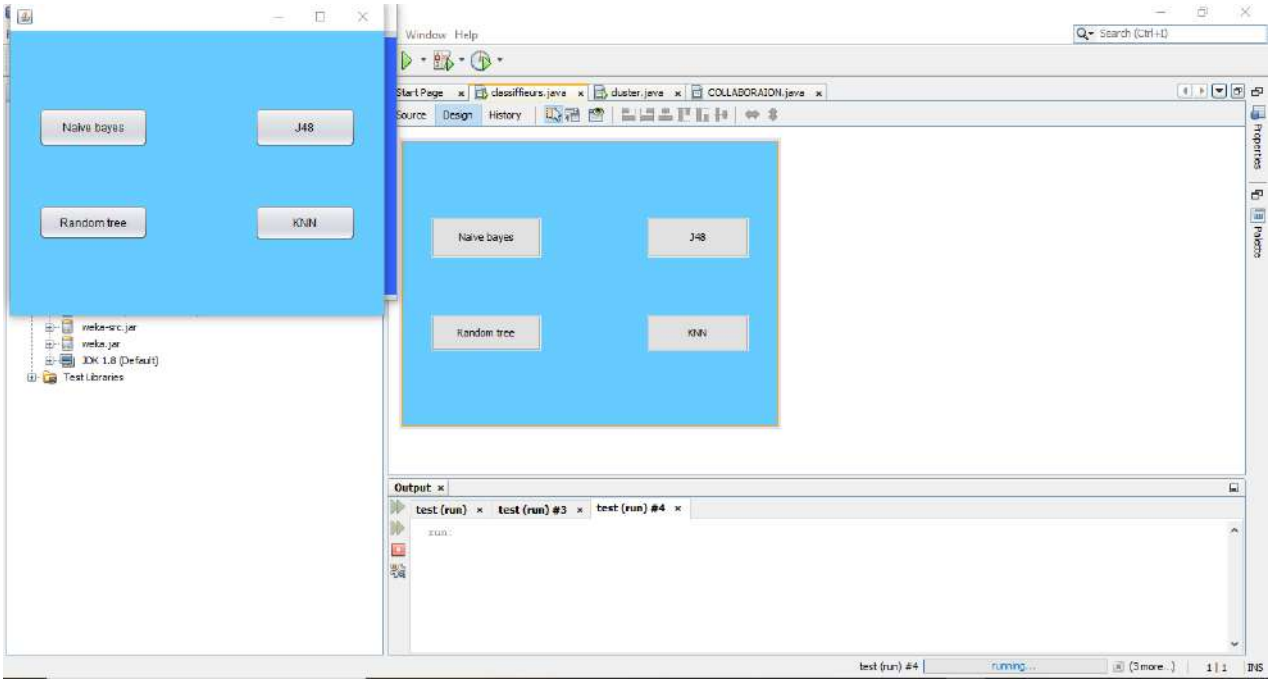
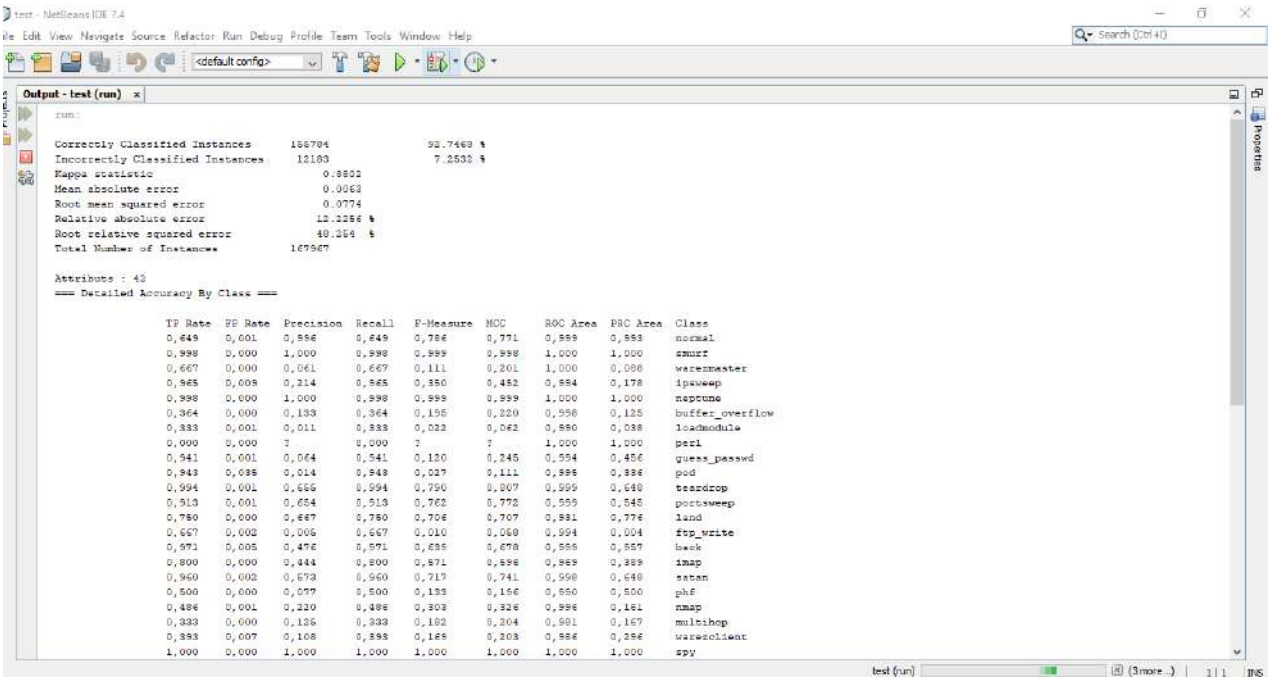


Figure 4.8 : interface des classifications

- **Choisir le classifieur naive bayes**



```
Correctly Classified Instances      186704               92.7469 %
Incorrectly Classified Instances     12103                7.2531 %
Kappa statistic                          0.8802
Mean absolute error                      0.0062
Root mean squared error                  0.0774
Relative absolute error                 12.1256 %
Root relative squared error             40.264  %
Total Number of Instances           198967

Attributes : 43
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0,649     0,001     0,956     0,649     0,786      0,771    0,989     0,983     normal
               0,998     0,000     1,000     0,998     0,999      0,998    1,000     1,000     smurf
               0,667     0,000     0,061     0,667     0,111      0,201    1,000     0,086     warezmaster
               0,965     0,009     0,214     0,965     0,350      0,452    0,994     0,178     ipsweep
               0,998     0,000     1,000     0,998     0,999      0,999    1,000     1,000     neptune
               0,364     0,000     0,133     0,364     0,195      0,220    0,998     0,125     buffer_overflow
               0,333     0,001     0,011     0,333     0,022      0,062    0,990     0,038     loadmodule
               0,000     0,000     ?         0,000     ?          ?        1,000     1,000     perl
               0,541     0,001     0,064     0,541     0,120      0,245    0,994     0,456     guess_passwd
               0,943     0,036     0,014     0,943     0,027      0,111    0,995     0,886     pod
               0,994     0,001     0,666     0,994     0,790      0,807    0,999     0,640     teardrop
               0,513     0,001     0,684     0,513     0,762      0,772    0,995     0,645     portsweep
               0,750     0,000     0,667     0,750     0,706      0,707    0,981     0,776     land
               0,667     0,002     0,006     0,667     0,010      0,060    0,994     0,004     ftp_write
               0,971     0,005     0,476     0,971     0,638      0,678    0,999     0,557     back
               0,800     0,000     0,444     0,800     0,571      0,596    0,989     0,389     imap
               0,960     0,002     0,673     0,960     0,717      0,741    0,998     0,648     satan
               0,500     0,000     0,077     0,500     0,133      0,196    0,990     0,500     phf
               0,486     0,001     0,220     0,486     0,303      0,326    0,996     0,181     nmap
               0,333     0,000     0,125     0,333     0,182      0,204    0,981     0,167     multihop
               0,393     0,007     0,108     0,393     0,169      0,203    0,986     0,296     warezclient
               1,000     0,000     1,000     1,000     1,000      1,000    1,000     1,000     spy
```

Figure 4.9: Résultats avec le classifieur **naïve** bayes

➢ **Evaluation sur les données de test :**

Notons que le temps pris par le processeur pour cet ensemble de test est 45,36 seconds, ce qui est considéré rapide.

➢ **Résumé des métriques :**

Métrique	Valeur	Pourcentage
Correctly Classified Instances	155784	92.7468 %
Incorrectly Classified Instances	12183	7.2532 %
Kappa statistic	**0.8802**	
Mean absolute error	0.0063	
Root mean squared error	0.0774	
Relative absolute error		12.2256 %
Root relative squared error		48.254 %
Total Number of Instances	167967	

Tableau 4.1 : Résultats de test avec le classifieur naïve bayes.

➢ **Précision détaillée par classe :**

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Normal	0,649	0,001	0,996	0,649	0,786	0,771	0,999	0,993
smuf	0,998	0,000	1,000	0,998	0,999	0,998	1,000	1,000
warezmaster	0,667	0,009	0,061	0,667	0,111	0,201	1,000	0,088
jspweep	0,965	0,000	0,214	0,965	0,350	0,452	0,994	0,178
neptune	0,998	0,000	1,000	0,998	0,999	0,999	1,000	1,000
Buffer_over	0,364	0,001	0,133	0,364	0,195	0,220	0,998	0,125
loadmodule	0,333	0,000	0,011	0,333	0,222	0,062	0,990	0,038
perl	0.000	0,001	?	0,000	?	?	1,000	1,000
Guess_passwd	0,941	0,035	0,064	0,941	0,120	0,245	0,994	0,456
pod	0,943	0,001	0,014	0,943	0,027	0,111	0,995	0,336

teardrop	0,994	0,001	0,655	0,994	0,790	0,807	0,999	0,648
portsweep	0,913	0,000	0,654	0,913	0,762	0,772	0,999	0,545
land	0,750	0,002	0,667	0,750	0,706	0,707	0,931	0,776
ftp_write	0,667	0,005	0,005	0,667	0,010	0,058	0,994	0,004
back	0,971	0,000	0,476	0,971	0,639	0,678	0,999	0,957
imap	0,800	0,002	0,444	0,800	0,571	0,596	0,969	0,389
satan	0,960	0,000	0,573	0,960	0,717	0,741	0,998	0,648
phf	0,500	0,001	0,077	0,500	0,133	0,196	0,990	0,500
nmap	0,486	0,000	0,220	0,486	0,303	0,326	0,996	0,161
multihop	0,333	0,007	0,125	0,333	0,182	0,204	0,981	0,167
warezclient	0,393	0,000	0,108	0,393	0,169	0,203	0,986	0,296
spy	1,000	0,007	1,000	1,000	1,000	1,000	1,000	1,000
Rootkit	0,500	0,000	0,002	0,500	0,004	0,030	0,977	0,002
Weighted avg	0,927	0,000	?	0,927	?	?	1,000	0,991

Tableau 4.2 : Résultat par classe avec le classifieur naïve bayes.

Nous constatons le taux élevé de la métrique Kappa (Dice) qui est de **0.8802** ce qui montre de bon choix de la part des données d'apprentissage. Aussi, un système de détection basé sur ce classifieur est considéré fiable et performant, car il y a que **0,1198** selon Dice qui reresente la fausse classification des données d'intrusion.

3.4. Test avec arbre décisionC.4.5 :

La figure suivante montre la sélection du classifieur C.4.5, nomé J48 dans Weka.

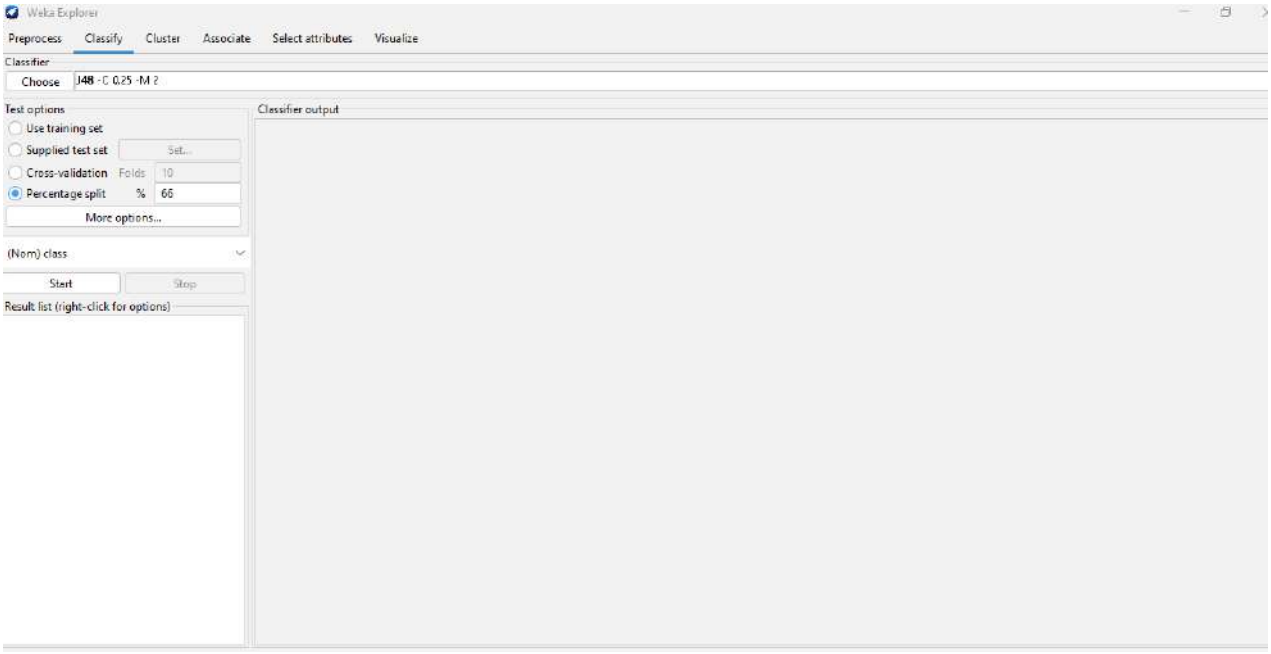


Figure 4.10: Sélection du classifieur C.4.5 (j 48).

3.4.1. Résultats de test C4.5 (j48) :

La figure 4.11 montre l'interface d'affichages des résultats de classification de données KDD.

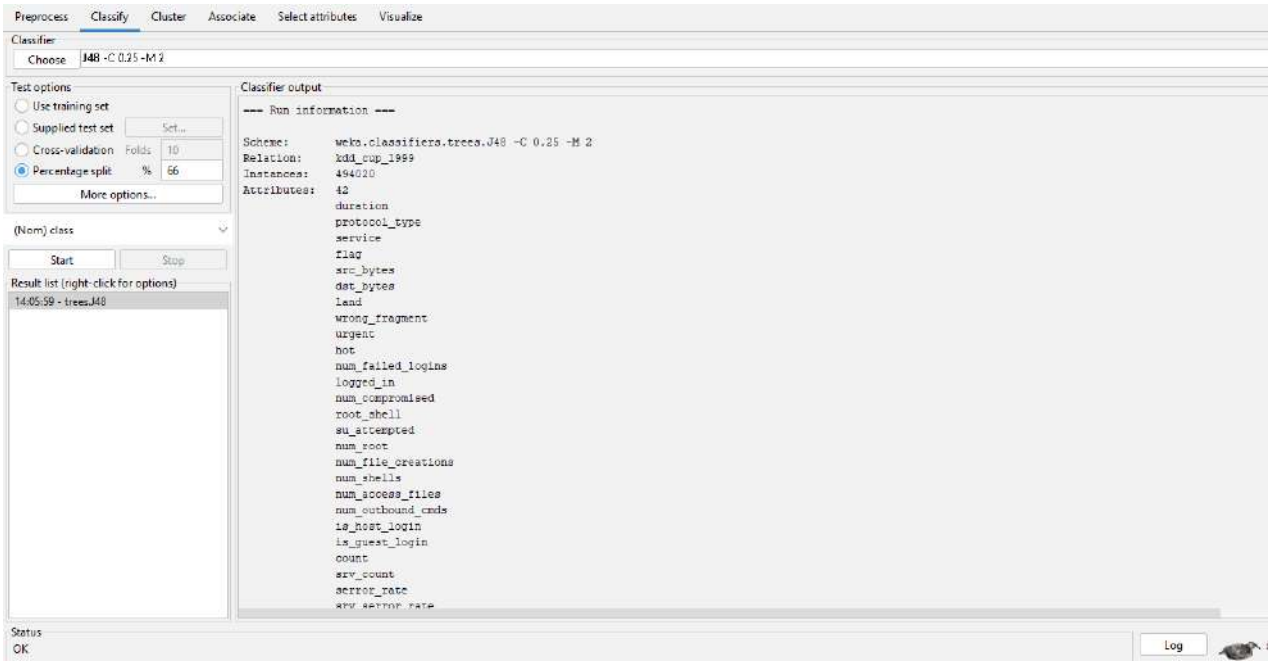


Figure 4.11: Résultats avec le classifieur C.4.5.

➢ **Evaluation sur l'ensemble de test :**

Notons que le temps pris par le processeur pour cet ensemble de test est 1,24 seconds, ce qui est considéré comme très rapide.

➢ **Résumé des métriques :**

Métrique	Valeur	Pourcentage
Correctly Classified Instances	167885	99,9512 %
Incorrectly Classified Instances	82	0,0488 %
Kappa statistic	**0.9992**	
Mean absolute error	0.0001	
Root mean squared error	0.0064	
Relative absolute error		0,1114 %
Root relative squared error		3.9606 %
Total Number of Instances	167967	

Tableau 4.3 : Résultats de test avec le classifieur C.4.5.

➢ **Précision détaillée par classe :**

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Normal	0,999	0,000	0,999	0,999	0,999	0,999	1,000	0, 999
smuf	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
warezmaster	1,000	0,000	0,375	1,000	0,545	0,612	1,000	0,500
jspweep	0,993	0,000	0,995	0,993	0,994	0,994	0,998	0,990
neptune	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
Buffer_over	0,636	0,000	0,875	0,636	0,737	0,746	0,790	0,538
loadmodule	0,000	0,000	0,000	0,000	0,000	-0,000	0,612	0,009
Perl	0.000	0,000	?	0,000	?	?	0,923	0,000
Guess_passwd	0,941	0,000	1,000	0,941	0,970	0,970	0,971	0,941
pod	0,943	0,000	1,000	0,943	0,970	0,971	0,971	0,943

teardrop	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
portsweep	0,983	0,000	0,994	0,983	0,989	0,989	0,997	0,979
land	0,750	0,000	1,000	0,750	0,857	0,866	0,875	0,750
ftp_write	0,000	0,000	0,000	0,000	0,000	-0,000	0,615	0,083
back	0,992	0,000	0,999	0,992	0,995	0,995	0,999	0,997
imap	0,200	0,000	0,500	0,200	0,286	0,316	0,600	0,100
satan	0,982	0,000	0,988	0,982	0,985	0,985	0,992	0,969
phf	1,000	0,000	0,667	1,000	0,800	0,816	1,000	0,667
nmap	0,959	0,000	0,959	0,959	0,959	0,959	0,986	0,947
multihop	0,000	0,000		0,000			0,667	0,026
warezclient	0,994	0,000	0,977	0,994	0,986	0,986	1,000	0,975
spy	0,000	0,000		0,000			0,998	0,002
Rootkit	0,000	0,000		0,000			0,670	0,000
Weighted avg	1,000	0,000	?	1,000	?	?	1,000	0,999

Tableau 4.4 : Résultat par classe avec le classifieur C.4.5.

Nous constantons le taux élevé de la métrique Kappa (Dice) qui est de 0.9992 ce qui montre de bon choix de la part des données d'apprentissage. Aussi, un système de dtection basé sur ce classifieur est considéré fiable et performant, car il y a que 0.0008 selon Dice qui reresente la fausse classification des données d'intrusion.

3.5 Test avec arbre aléatoire (random tree)

La figure suivante montre la sélection du classifieur **arbre aléatoire**, nomé **random tree** dans Weka.

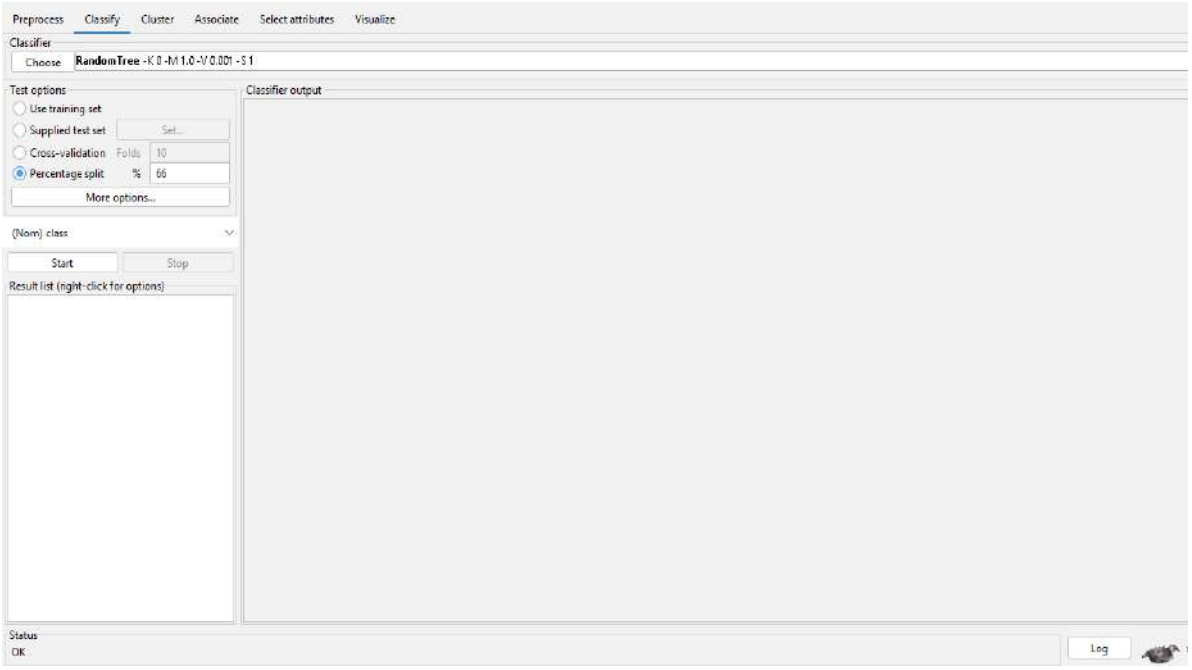


Figure 4.12: Sélection du classifieur arbre aléatoire (random tree).

3.5.1 Résultats de test :

La figure 4.13 montre l'interface d'affichages des résultats de classification de données KDD.

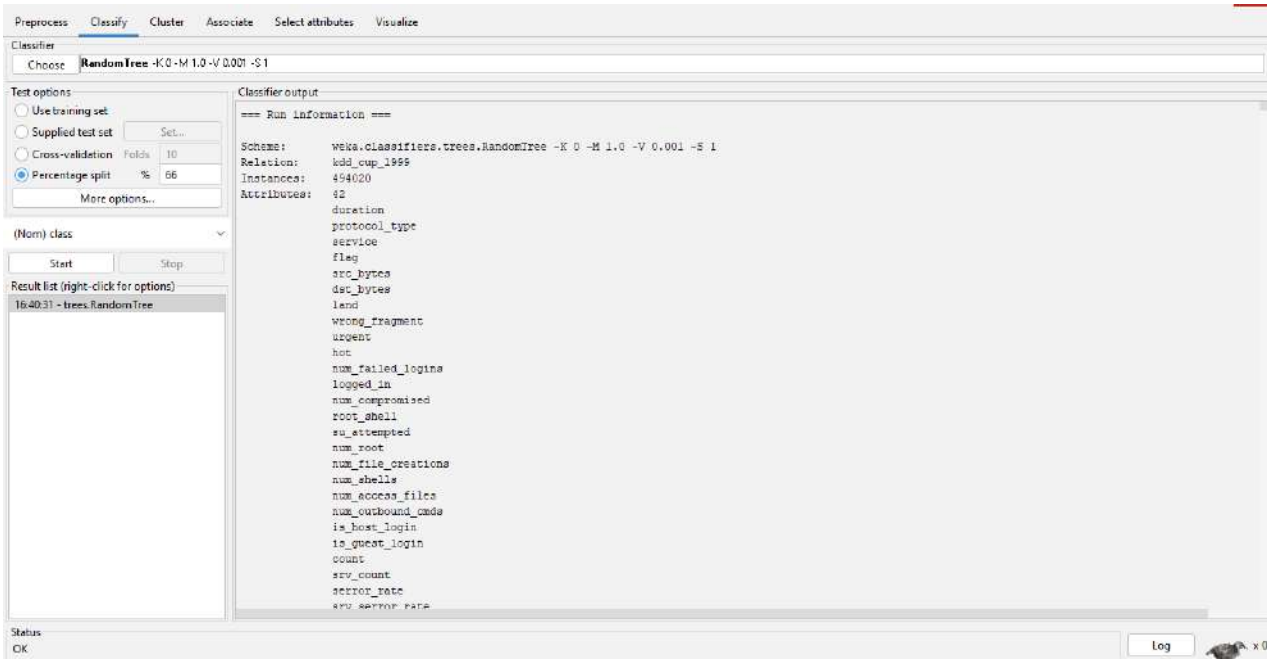


Figure 4.13: Résultats avec le classifieur arbre aléatoire (random tree).

- **Evaluation sur les données de test :**

Notons que le temps pris par le processeur pour cet ensemble de test est 1,38 seconds, ce qui est considéré rapide.

- **Résumé des métriques :**

Métrique	Valeur	Pourcentage
Correctly Classified Instances	167854	99.9327 %
Incorrectly Classified Instances	113	0.9327 %
Kappa statistic	**0.9989**	
Mean absolute error	0.0001	
Root mean squared error	0.0076	
Relative absolute error		0.1158%
Root relative squared error		4.7455%
Total Number of Instances	167967	

Tableau 4.5 : Résultats de test avec le classifieur.arbre aléatoire **(random tree)**

- **Précision détaillée par classe :**

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Normal	0,999	0,000	0,999	0,999	0,999	0,999	0,999	0,998
smuf	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
warezmaster	0,667	0,000	0,667	0,667	0,667	0,667	0 ,833	0,444
jspweep	0,988	0,000	0,986	0,988	0,987	0,987	0,994	0,975
neptune	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
Buffer_over	0,364	0,000	0,500	0,364	0,421	0,426	0,682	0,182
loadmodule	0,000	0,000	0,000	0,000	0,000	-0,000	0,500	0,000
perl	0.000	0,000	0,000	0,000	0,000	-0,000	0,500	0,000
Guess_passwd	0,941	0,000	1,000	0,941	0,970	0,970	0,971	0,941
pod	0,943	0,000	0,953	0,943	0,948	0,948	0,971	0,909
teardrop	0,988	0,000	0,977	0,988	0,983	0,983	0,994	0,971
portsweep	0,983	0,000	0,997	0,983	0,990	0,990	0,993	0,983

land	0,750	0,000	0,000	0,000	0,750	0,750	0,875	0,563
ftp_write	0,667	0,000	0,667	0,667	0,667	0,667	0,833	0,444
back	0,999	0,000	0,999	0,999	0,999	0,999	0,999	0,999
imap	0,200	0,000	1,000	0,200	0,333	0,447	0,700	0,300
satan	0,986	0,000	0,974	0,986	0,980	0,980	0,993	0,961
phf	0,500	0,000	1,000	0,500	0,667	0,707	0,750	0,500
nmap	0,919	0,000	0,932	0,919	0,925	0,925	0,959	0,880
multihop	0,333	0,000	0,333	0,333	0,333	0,333	0,667	0,111
warezclient	0,962	0,000	0,974	0,962	0,968	0,968	0,981	0,937
spy	0,000	0,000		0,000			0,500	0,000
Rootkit	0,500	0,000	0,667	0,500	0,571	0,577	0,750	0,333
Weighted avg	0,999	0,000	?	0,999	?	?	1,000	0,999

Tableau 4.6 : Résultat par classe avec le classifieur arbre aléatoire **(random tree)**

.Nous constantons le taux élevé de la métrique Kappa (Dice) qui est de 0.9989 ce qui montre de bon choix de la part des données d'apprentissage. Aussi, un système de dtection basé sur ce classifieur est considéré fiable et performant, car il y a que 0.0011 selon Dice qui reresente la fausse classification des données d'intrusion.

3.6 Test avec le classifieur k-nn (IBK):

La figure suivante montre la sélection du classifieur des plus proche voisins k-nn, nomé IBK dans Weka.

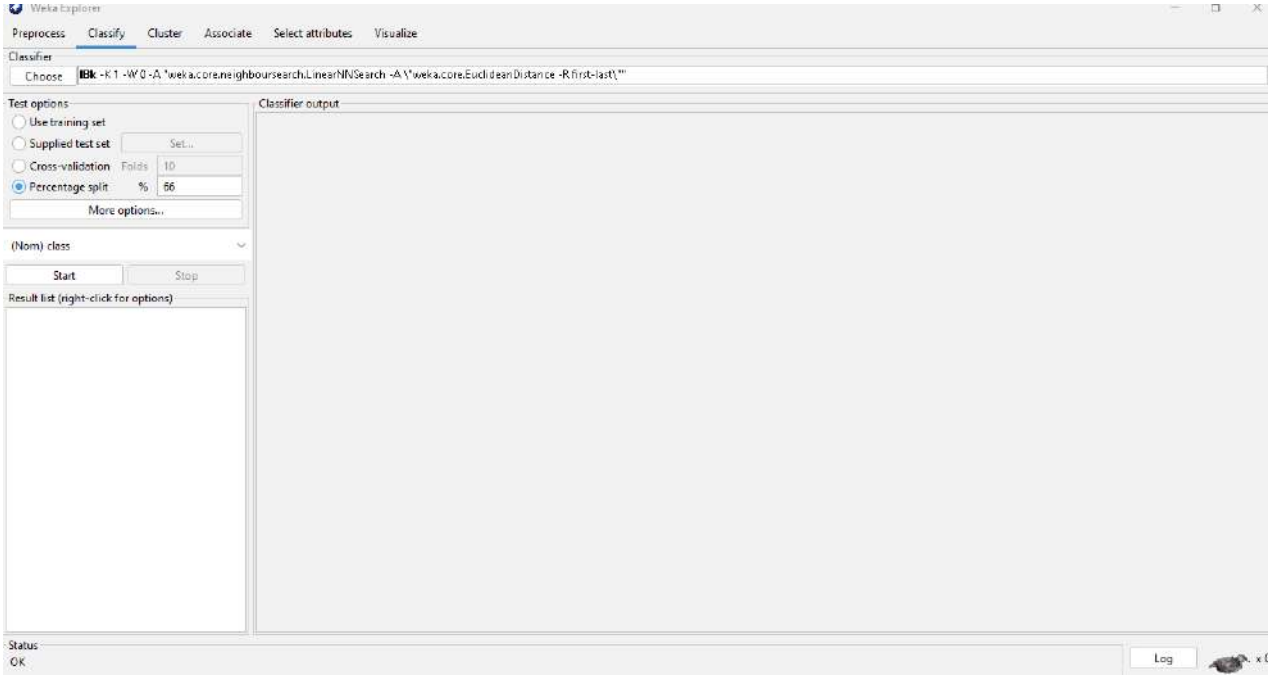


Figure 4.14: Sélection du classifieur K-nn.

3.6.1 Résultats de test knn (ibk):

La figure 4.15 montre l'interface d'affichages des résultats de classification de données KDD.

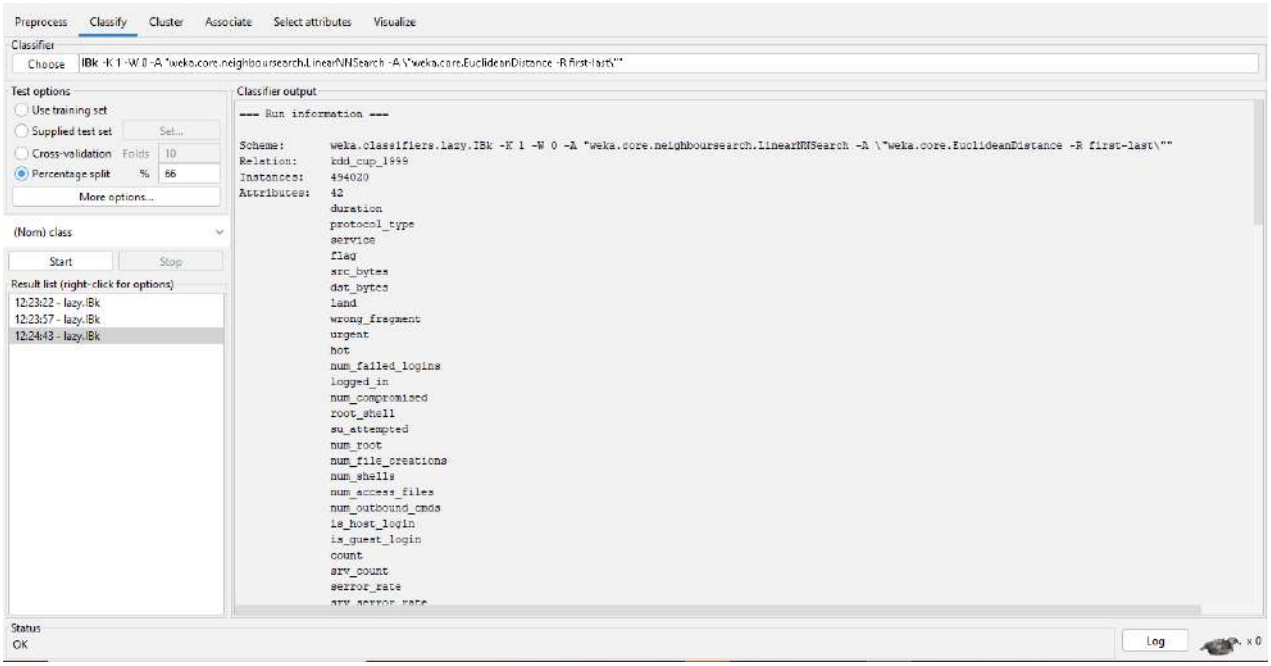


Figure 4.15: Résultats avec le classifieur k-nn.

- **Evaluation sur les données de test :**

Notons que le temps pris par le processeur pour cet ensemble de test est 59.55 seconds, ce qui est considéré rapide.

- **Résumé des métriques :**

Métrique	Valeur	Pourcentage
Correctly Classified Instances	167869	99.9417 %
Incorrectly Classified Instances	98	0.0583 %
Kappa statistic	**0.999**	
Mean absolute error	0.0001	
Root mean squared error	0.0071	
Relative absolute error		0.1024%
Root relative squared error		4.4419 %
Total Number of Instances	167967	

Tableau 4.7 : Résultats de test avec le classifieur k-nn.

- **Precision détaillée par classe:**

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
Normal	0,999	0,000	0,999	0,999	0,999	0,998	1,000	0,998
smuf	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
warezmaster	0,667	0,000	0,500	0,667	0,571	0,577	0 ,953	0,333
jspweep	0,979	0,000	0,998	0,979	0,988	0,988	0,997	0,978
neptune	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
Buffer_over	0,636	0,000	0,700	0,636	0,667	0,667	0,949	0,446
loadmodule	0,333	0,000	0,333	0,333	0,333	0,333	0,906	0,111
perl	1.000	0,000	0,250	1,000	0,400	0,500	1,000	0,250
Guess_passwd	0,941	0,000	1,000	0,941	0,970	0,970	0,992	0,941
pod	0,943	0,000	0,976	0,943	0,959	0,959	0,992	0,926
teardrop	0,997	0,000	1,000	0,997	0,999	0,999	1 ,000	0,997

portsweep	1,000	0,000	0,994	1,000	0,997	0,997	1,000	0,998
land	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000
ftp_write	0,667	0,000	0,500	0,667	0,571	0,577	0,935	0,333
back	0,981	0,000	0,984	0,981	0,982	0,982	0,997	0,975
imap	0,600	0,000	1,000	0,600	0,750	0,775	0,944	0,600
satan	0,992	0,000	0,998	0,992	0,995	0,995	0,999	0,991
phf	0,500	0,000	1,000	0,500	0,667	0,707	0,930	0,500
nmap	0,986	0,000	0,961	0,986	0,973	0,973	0,998	0,964
multihop	0,000	0,000	0,000	0,000	0,000	-0,000	0,859	0,000
warezclient	0,977	0,000	0,963	0,977	0,970	0,970	0,997	0,943
spy	0,000	0,000	?	0,000	?	?	0,859	0,000
Rootkit	0,250	0,000	1,000	0,250	0,400	0,500	0,854	0,250
Weighted avg	0,999	0,000	?	0,999	?	?	1,000	0,999

Tableau 4.8 : Résultat par classe avec le classifieur K-nn.

Nous constatons le taux élevé de la métrique Kappa (Dice) qui est de 0.999 ce qui montre de bon choix de la part des données d'apprentissage. Aussi, un système de détection basé sur ce classifieur est considéré fiable et performant, car il y a que 0.001 selon Dice qui représente la fausse classification des données d'intrusion.

4. Discussion des résultants

Selon les résultats obtenus avec les quatre classifieurs, on constate des taux élevés des métriques de performances, et des taux faibles des métriques d'erreur. Ceci implique le bon choix de la sélection des données d'apprentissage et aussi des classifieurs utilisés.

Cependant, nous remarquons une légère avancée du classifieur C 4.5 par rapport au classifieur naïve bayes et le classifieur arbre aléatoire et le classifieur KNN En effet, le classifieur C 4.5 a enregistré un indice de Dice = 0.9992, ce qui est élevé par rapport l'indice de Dice pour le classifieur KNN, qui a enregistré 0.999 et de Dice pour le classifieur arbre aléatoire qui a enregistré 0,9989 et de Dice pour le classifieur naïve bayes qui a enregistré 0,8802.

5.Test avec k-means :

La figure suivante montre la sélection du cluster k-means .

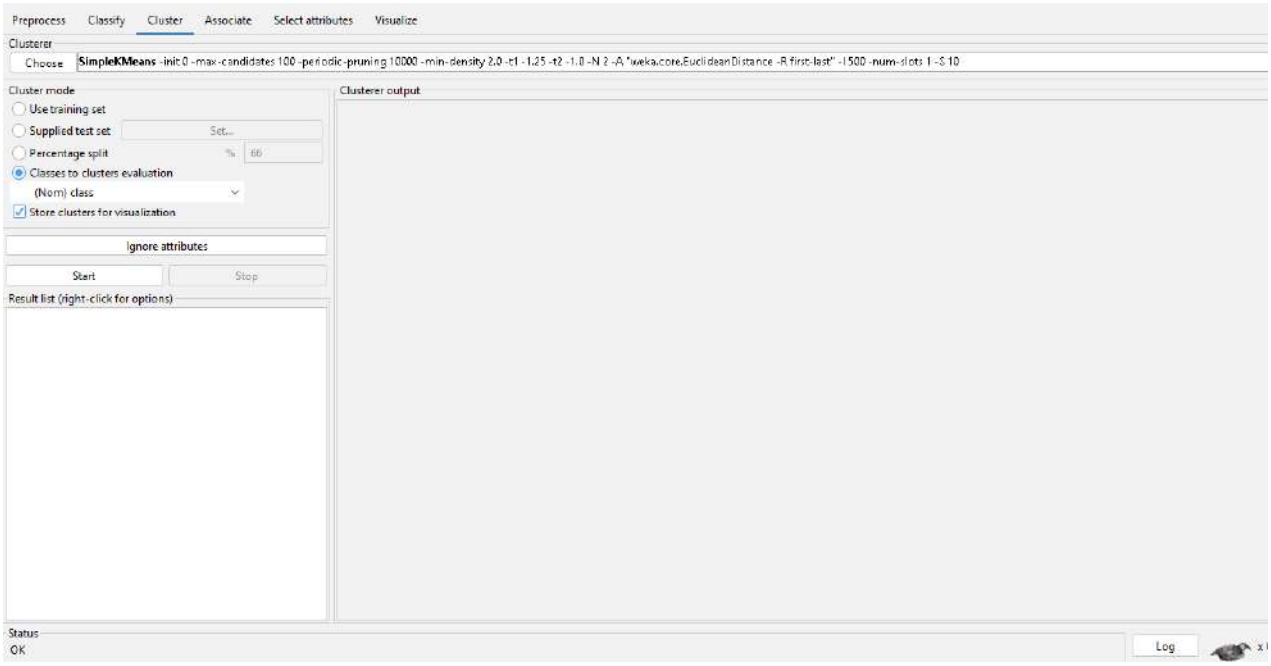


Figure 4.16: Sélection du clustering.

5.1 Résultats de test k-means :

La figure 4.17 montre l'interface d'affichages des résultats de clustring de données KDD.

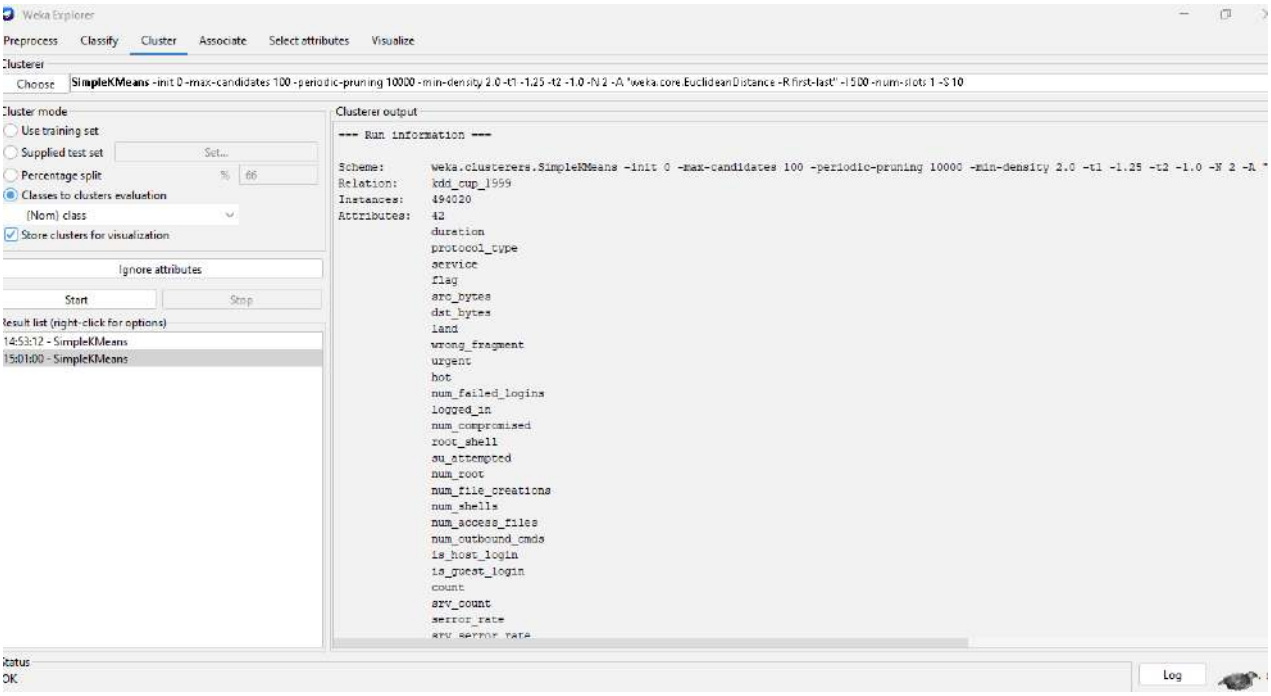


Figure 4.17: Résultats avec le clustering k-means.

Il y a une autre méthode avec java sous NetBeans.

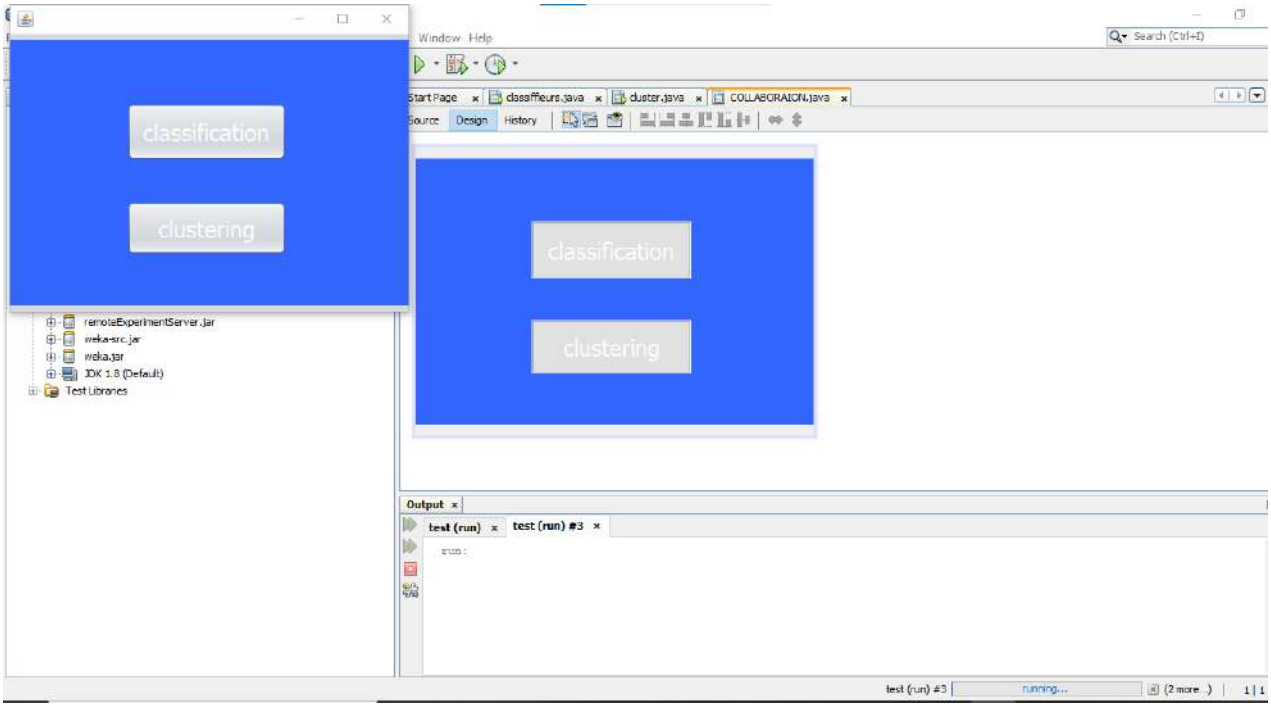


Figure 4.18 : interface d'accueil

- **Click sur le Button clustering.**

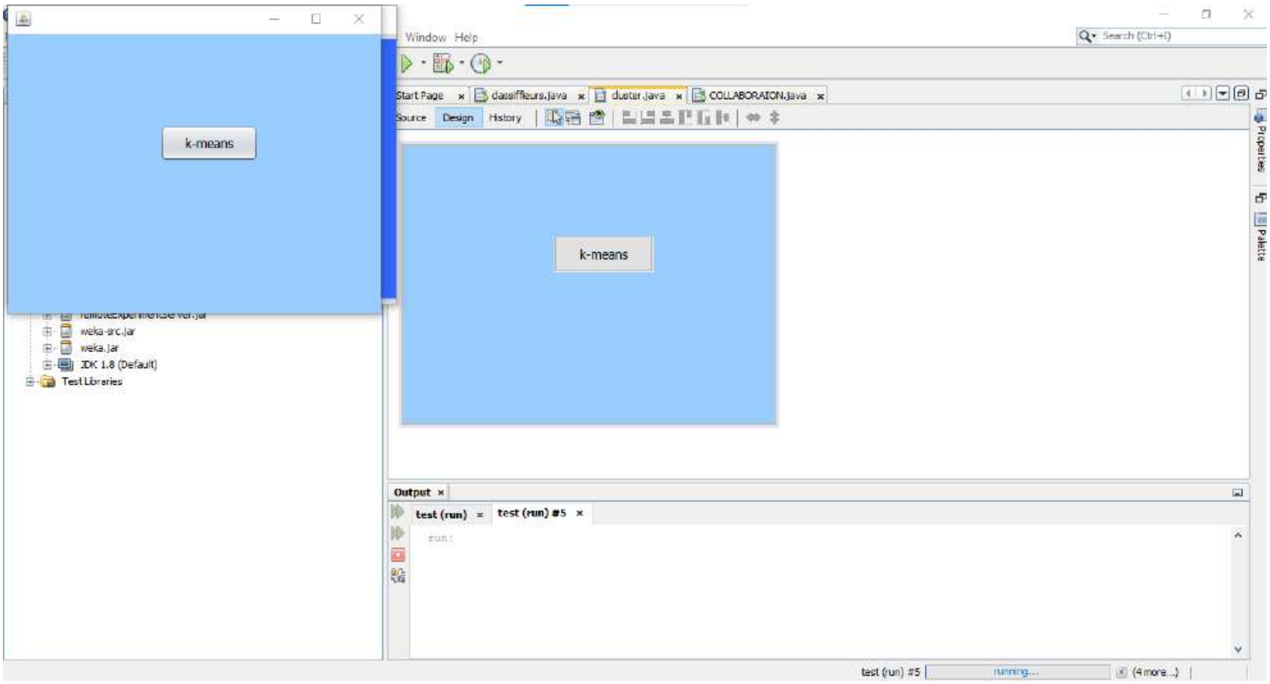


Figure 4.19 : interface de clustering

- **Choisir le clustering k-means.**

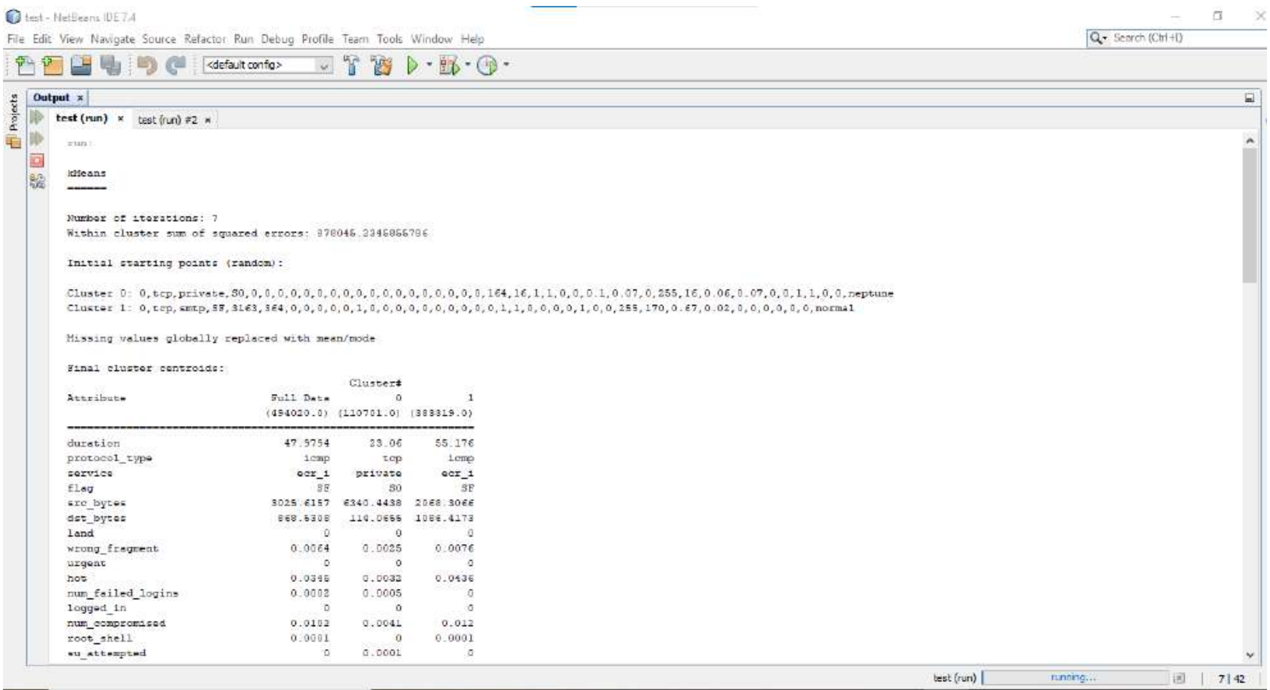


Figure 4.20: Résultats avec le clustering k-means.

❖ **Remarque :**

On peut aussi utiliser le NetBeans dans tous les classificateurs. Et nous avons utilisé le NetBeans pour les personnes qui ne connaissent pas le weka

➢ **Evaluation sur les données de test :**

Notons que le temps pris par le processeur pour cet ensemble de test est 5.86 seconds, ce qui est considéré rapide.

➢ **Résumé des métriques :**

Métrique	Valeur	Pourcentage
Clustred Instances	Cluster 0 :110701 Cluster 1 :383319	Cluster 0 :22% Cluster 1 :78%
Incorrectly Classified Instances	106029	21.4625%
Cluster sum of squared error	772016.2345835345	
Number of Iterations	7	
Total Number of Instances	167967	

Tableau 4.9 : Résultats de test avec le cluster k-means.

➢ **Precision détaillée par classe :**

Class	Cluster 0	Cluster 1
Normal	596	96681
smuf	0	280790
warezmaster	1	19
jspweep	87	1160
neptune	107201	0
Buffer_over	2	28
loadmodule	0	9
perl	1	2

Guess_passwd	51	2
pod	0	264
teardrop	95	884
portsweep	1038	2
land	21	0
ftp_write	0	8
back	0	2203
imap	7	5
satan	1484	105
phf	0	4
nmap	103	128
multihop	0	7
warezclient	12	1008
spy	1	1
Rootkit	1	9

Cluster 0 : Neptune
Cluster 1 : smurf

Tableau 4.10 : Résultat par classe avec le clustering k-means

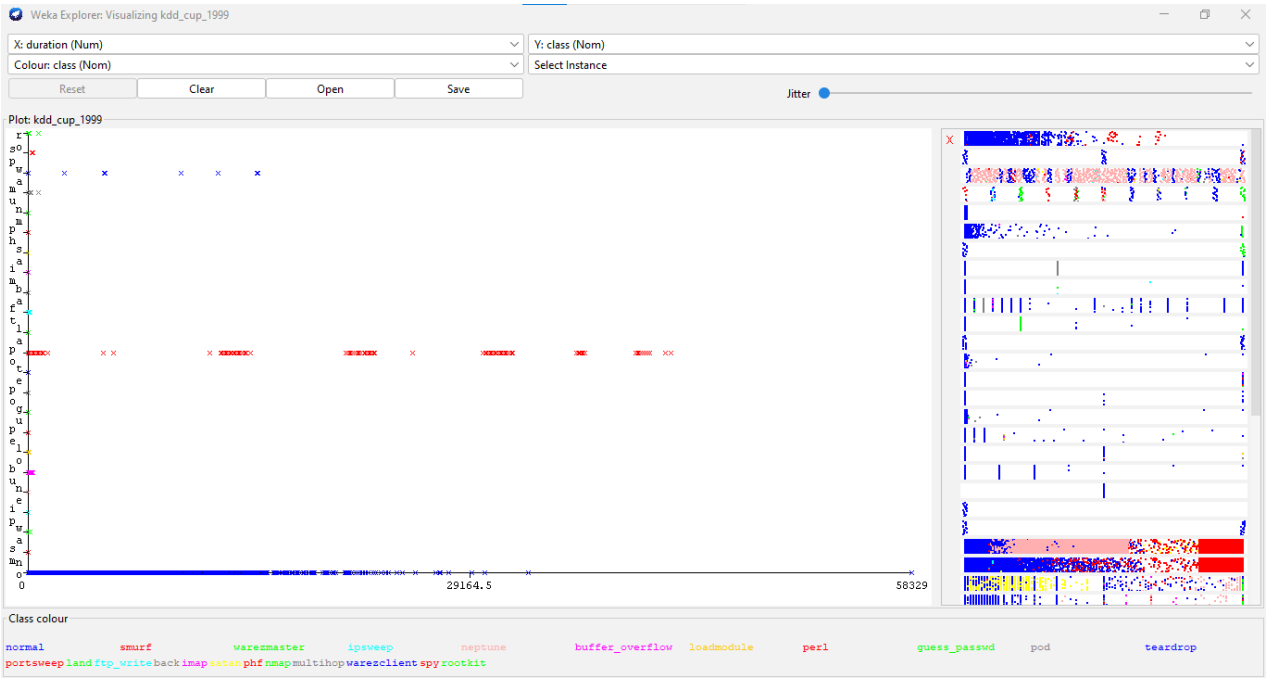


Figure 4.21 : visualisation des erreurs de cluster k- means.

Nous constatons le nombre d'erreurs ou carré à l'intérieur du cluster élevé qui et de 106029 ce qui montre de mal choix de la part des données d'apprentissage.

6. Conclusion :

Dans ce dernier chapitre, nous avons présenté la partie expérimentale de notre travail, qui consiste à tester les données d'intrusion de la base de données KDD. Nous avons testé les quatre classifieurs naïve bayes et l'arbre de décision et les arbres aléatoires et le KNN et cluster k-means. En présentant différentes métriques de performance dont l'indice de Dice (Kappa) et taux des métriques d'erreur. Dans les classifieurs enregistrent une bonne performance par contre k-means non.

Conclusion générale

Conclusion générale

Dans ce mémoire, nous nous sommes intéressées à la détection d'intrusions partir des données du trafic réseau. Ces données sont fournies dans la base de données KDD.

Plus précisément, nous avons abordé la méthode de l'algorithme naïve bayes et l'arbre décision et l'arbre aléatoire et k-plus proche voisin et le cluster k-means.

Et pour ne pas commencer de zéro, nous avons utilisé la bibliothèque Weka qui est une collection d'algorithmes d'apprentissage automatique pour la fouille des données et java sous NetBeans.

Ce travail avait pour but, de comparer les précisions de détection des classifieurs avec les données d'intrusion de la base KDD.

Les expérimentations que nous avons menées et les résultats que nous avons obtenus ont montré que les classifieurs naïve bayes et l'arbre décision et l'arbre aléatoire et k-plus proche voisin.et k-means sont proches en termes de précision de détection.

Cependant le C 4.5 est extrêmement plus rapide que la naïve bayes et l'arbre décision et l'arbre aléatoire et k-plus proche voisin.

En perspective à ce travail, il est possible de tester d'autres classifieurs, supervisés et non supervisés, en utilisant la même base de données.

Bibliographie

Bibliographie

[1] : Liran LERMAN, Les systèmes de détection d'intrusion basés sur du machine Learning, UNIVERSITÉ LIBRE DEBRUXELLES.

[2] : Jonathan Krier, Les systèmes de détection d'intrusions, document réalisé dans le cadre d'un sujet d'initiation à la recherche en Master, publié sur le site web Developpez.com, 21 juillet 2006.

[3] : Melle AISSAOUI Sihem, Apprentissage automatique et sécurité des systèmes d'information avec l'Application : Un système de détection d'intrusion basé sur les Séparateurs àVasteMarge (SVM), Diplôme de Magister, Université d'OranEs-senia, 2007-2008

[4] : Melle Asma CHIKH et Melle Amina DJENNANE, Sécurité d'une application Web à l'aide d'un système de détection d'intrusions comportementale, Diplôme de Master, Université Abou Bakr Belkaid– Tlemcen, 2011-2012.

[5] :https://fr.wikipedia.org/wiki/Logiciel_espion.html.

[6] :https://fr.wikipedia.org/wiki/Spam.html.

[7] : Mme. BOUKHLOUF Djemaa, Une approche à base d'agents mobiles pour la sécurité des systèmes d'informations sur le web, Thèse de Doctorat, UNIVERSITE MOHAMED KHIDERBISKRA,2016.

[8] : Cisco et la sécurité, Dossier de presse, Novembre2004.

[9] :https://www.guill.net

[10] : https://fr.wikipedia.org/wiki/ Système_de_détection_d'intrusion.html

[11] : M. TOUATI Azeddine, Détection d'intrusions dans les réseaux LAN : Installation et configuration de l'IDS-SNORT, Diplôme de Master, Université A /Mira de Bejaïa , 2015-2016.

[12] : K. BELKHATMI et O. BENAMARA, Mise en place d'un système de détection et de prévention d'intrusion, mémoire de master 02 Université A/Mira de Béjaïa, 2016.

[13] : S. AGGOUN, S. BELKACEM, Mise en oeuvre d'une solution de sécurité basée sur les IDS Cas d'étude : entreprise Cevital, Mémoire de Master en Informatique, Université ABDERAHMANE MIRA de Bejaia, 2013.

[14] : J. TIMMIS. Artificial immune systems: A novel data analysis technique inspired by the immune network theory, 1999.

[15] : Microsoft Experiences, Tout savoir sur l'Intelligence Artificielle, (consulté le 01/03/2022), disponible sur : https://experiences.microsoft.fr/business/intelligenceartificielle-ia-business/comprendreutiliser-intelligence-artificielle/

[16] A. L. Samuel, "Some studies in machine learning using the game of checkers,IBM Journal of research and development, vol. 3, no. 3, pp. 210–229, 1959.

[17] T. M. Mitchell, "Machine learning, volume 1 of 1," 1997.

[18] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," Neural computation, vol. 18, no. 7, pp. 1527–1554, 2006.

[19] A. Géron, Hands-on machine learning with Scikit-Learn, Keras, and Tensor Flow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media, 2019.

[20] I. Vasilev, D. Slater, G. Spacagna, P. Roelants, and V. Zocca, Python Deep Learning: Exploring deep learning techniques and neural network architectures with Pytorch, Keras, and Tensor-Flow. Packt Publishing Ltd, 2019.

[21] L.-P. Chen, "Mehryar mohri, afshin rostamizadeh, and ameet talwalkar : Foundations of machine learning," 2019.

[22] J. Friedman, T. Hastie, and R. Tibshirani, The elements of statistical learning. Springer series in statistics New York, 2001, vol. 1, no. 10.

[23] P. Kardi Teknomo, "K-Mean Clustering algorithm."

[24] W. S. McCulloch et W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115-133, 1943.

[25] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[26] A. K. Jain, J. Mao, et K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31-44, 1996.

[27] :P. A. Jaskowiak et R. J. G. B. Campello, « Comparing Correlation Coefficients as Dissimilarity Measures for Cancer Classification in Gene Expression Data », sur http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.208.993, Brazilian Symposium on Bioinformatics (BSB 2011) (consulté le 16 mai 2022), p. 1–8 .

[28]: MIT Lincoln Labs, 1998 DARPA Intrusion Detection Evaluation. Available on: http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/ index.html, February 2008.

[29]: : Srinivas Mukkamala&& all "Intrusion detection using an ensemble of intelligent paradigms", Journal Network and Computer Applications 28 (2005), 167-182.

[30] D. Coomans et D.L. Massart, « Alternative k-nearest neighbour rules in supervised pattern recognition: Part 1. k-Nearest neighbour classification by using alternative voting rules », *AnalyticaChimicaActa*, vol. 136, 1982, p. 15–27

[31] https://www.c-s.fr/Eclipse-bien-plus-qu-un-environnement dedeveloppement_a780.html.

[32] :(en) Ian H. Witten, Eibe Frank, et Mark A. Hall, *Data Mining: Practical machine learning tools and techniques*,3^{e}édition, Morgan Kaufmann, 2011 (ISBN 978-0-1237-4856-0), 629 pages [présentationenligne]

[33]:(en) G. Holmes, A. Donkin and I.H. Witten, « Weka: A machine learning workbench », Proc Second Australia and New Zealand Conference on Intelligent Information Systems, Brisbane, Australia, 1994 (consulté le 1 juin 2022) [PDF]

[34] :(en) S.R. Garner, S.J. Cunningham, G. Holmes, C.G. Nevill-Manning, and I.H. Witten, « Applying a machine learning workbench: Experience with agricultural databases », Proc Machine Learning in Practice Workshop, Machine Learning Conference, Tahoe City, CA, USA, 1995 (consulté le 1 juin 2022), p. 14–21 [PDF].

[35] :(en) P. Reutemann, B. Pfahringer and E. Frank, « Proper: A Toolbox for Learning from Relational Data with Propositional and Multi-Instance Learners », 17th Australian Joint Conference on Artificial Intelligence (AI2004), Springer-Verlag, 2004 (consulté le 1 juin 2022).