MASTER EN SCIENCES INFORMATIQUES

Machine Learning et Sécurité

Detomal Remadji, Trésor

Award date:
2019

Awarding institution:
Universite de Namur

Université de Namur
Faculté d'informatique
Année académique 2017–2018

Machine Learning et Sécurité

Detomal Remadji Trésor

Promoteur : _____ (Signature pour approbation du dépôt - REE art. 40)
Jean Noel Colin

Mémoire présenté en vue de l'obtention du grade de
Master en Sciences Informatiques.

Résumé

Le domaine de la cybersécurité est un domaine très vaste qui vise essentiellement à apporter une protection à des systèmes informatisés. Le Machine Learning est l'une des techniques servant à aider dans les prises de décisions face aux différents problèmes liés à la sécurité informatique. L'objectif clé de l'utilisation du Machine Learning est d'apporter une solution à des problématiques telles que l'authentification via la biométrie, la détection d'intrusion, la détection des botnets, la détection des tunnels DNS, la détection des DGA, la détection des injections SQL, la détection des spams ainsi que la corrélation d'alertes. Sur base des jeux de données [17], nous avons proposé une amélioration de l'implémentation [22] pour la détection des spams en utilisant le **Pipeline** afin de combiner le **GridSearchCV** afin de trouver les hyper-paramètres du Naïve Bayes. Cette amélioration a permis d'obtenir une précision de 99% ainsi qu'un rappel de 99%. Nous allons également explorer les autres algorithmes du Machine Learning tels que le **S**upport **V**ector **M**achine (SVM), l'arbre de décision, la **R**égression **L**ogistique (RL) afin d'analyser les résultats.

Abstract

The field of cybersecurity is a very broad field that aims essentially at providing protection to computerized systems. Machine Learning is one of the techniques used to assist decision-making in the face of various IT security issues. The key objective of using Machine Learning is to provide solutions to issues such as authentication via biometrics, intrusion detection, botnet detection, DNS tunnel detection, detection DGAs, SQL injection detection, spam detection and alert correlation. Based on [17] datasets, we proposed an implementation enhancement [22] for spam detection using the **Pipeline** to combine the **GridSearchCV** to find the hyper-parameters of the Naïve Bayes. This improvement resulted in precision of 99% and recall of 99% . We will also explore other Machine Learning algorithms such as the Support Vector Machine (SVM), the decision tree, the Logistic Regression (RL) to analyze the results.

Remerciements

Je tiens à exprimer ici tous mes remerciements ainsi que toute ma gratitude à toutes les personnes sans lesquelles ce travail n'aurait pas été mené à terme.

En premier lieu, mes remerciements vont à Mr Jean-Noel Colin pour m'avoir guidé et m'avoir orienté dans mes travaux du présent mémoire.

En second lieu, je tiens à remercier très sincèrement mon père pour son soutient et tous les membres de ma famille pour leur confiance et leurs encouragements pendant mes études.

Table des matières

Glossaire

DGA : **D**omain **G**enerating **A**lgorithm

DNS : **D**omain **N**ame **S**ystem

FN : **F**aux **n**égatifs

FP : **F**aux **p**ositifs

IDS : **I**ntrusion **D**etection **S**ystem

KNN : **K**-**N**earest **N**eighbors

LR : **L**inear **R**egression

NB : **N**aive **B**ayes

RL : **R**égression **L**ogistique

SQL : **S**tructured **Q**uery **L**anguage

SVM : **S**upport **V**ector **M**achine

URL : **U**niform **R**esource **L**ocator

VN : **V**rais **n**égatifs

VP : **V**rais **p**ositifs

Introduction

Le Machine Learning a fait ses preuves dans beaucoup de domaines divers et variés. Le Machine Learning a aidé dans le secteur financier notamment dans les banques et les assurances à prédire si un demandeur est solvable, à détecter la fraude par carte de crédit, à trouver des tendances prometteuses sur le marché boursier, etc.

Le Machine Learning peut également aider à diagnostiquer et détecter les problèmes tels que les cancers. Le Machine Learning permet également de faire la reconnaissance des images et la reconnaissance vocale. La reconnaissance des images est également utilisée dans le cadre de l'identification par empreinte digitale qui sera abordée dans ce rapport. La reconnaissance vocale permet quant à elle de communiquer des actions par voie vocale à un système informatisé tel que **SIRI** utilisé dans les smartphones.

Dans le domaine de la cybersécurité, les différentes méthodes du Machine Learning ont été utilisées à travers les problèmes de sécurité tels que la détection de spams, la détection d'intrusion, la corrélation d'alertes, l'authentification par Keystroke ainsi que l'identification par empreinte digitale et aussi d'autres d'utilisations.

A travers ce mémoire nous souhaitons présenter le Machine Learning et ses différents aspects ainsi que son fonctionnement d'une part et ferons un état de l'art des différents domaines de la cybersécurité où les différents algorithmes du Machine Learning ont été appliqués d'autre part.

PREMIÈRE PARTIE

Machine Learning

Chapitre 1

Introduction

Avec l'arrivée de l'internet, les machines et les objets se sont développés de façon exceptionnelle en générant des flux de données. Ces flux de données peuvent être exploités via des outils mathématiques et de statistiques afin de les analyser et d'en faire des déductions. Ces outils ne s'avèrent pas être très adéquats en raison de l'accroissement des données et l'expansion technologique pour faire des prédictions. En effet, ces techniques n'évoluent pas et présentent des limites sur la représentation des données (par exemple faire une recommandation d'une chanson à un utilisateur). Il faut donc un outil permettant d'apprendre par lui même sur base des jeux de données (dataset) fournis et de s'améliorer automatiquement d'où l'apparition du Machine Learning ou l'apprentissage automatique.

Il existe plusieurs définitions du Machine Learning. Selon **Arthur Samuel** l'un des pionniers de l'intelligence artificielle en 1959, *"le Machine Learning est défini comme le champ de l'étude visant à donner la capacité à un ordinateur (une machine), d'apprendre par lui même sans y être programmé via des algorithmes d'apprentissages"* [6]. La définition du Machine Learning a considérablement évolué et une définition plus récente a été apportée par **Tom Mitchell** de l'université de Carnegie Mellon qui selon lui, *"on dit qu'un programme apprend de l'expérience E par rapport à une tâche T et une mesure de performance P si sa performance sur T, mesurée par P, s'améliore avec l'expérience E"* [97].

Le Machine Learning est une branche de l'intelligence artificielle qui se base sur des algorithmes de *Data-Mining* qui ont montré leur limite en terme de performance dû au manque de données. Il fait référence à l'analyse et à l'implémentation de méthodes qui permettent à une machine d'apprendre grâce à un processus d'apprentissage sur des jeux de données relativement importants, et d'exécuter des tâches où il est difficile voir impossible de remplir par des moyens algorithmiques classiques.

Le but de l'utilisation du Machine Learning face à un problème est de faire la classification, la régression ou le regroupement sur base de la distance ou de la densité.
Le Machine Learning est utilisé pour créer des modèles de prédiction à partir des données. Le résultat de la prédiction dans le cadre de la classification est de type discret car on cherche à prédire la valeur d'une classe existante. Les cas concrets de l'utilisation du Machine Learning dans le but de faire de la classification sont : la détection de spams, la détection d'intrusion, etc.

Par opposition à des problèmes de type classification qui produisent des résultats qualitatifs, les problèmes de type régression quant à eux prédisent des résultats de type continus sur base des données observées. La régression doit estimer une fonction qui fait correspondre les données en entrée avec la sortie. Ils sont utilisés pour prédire sur base des données représentatives de chaque situation par exemple pour prédire la consommation d'énergie, le prix des maisons, estimer le prix du stock, etc [93].

Le clustering vise à regrouper les données fournies en classes. Il est souvent considéré comme le synonyme de l'apprentissage non-supervisé. Il décompose les données en plusieurs sous ensembles en s'assurant que les éléments dans chaque sous-ensembles soient les plus homogènes que possible vue que les données ne contiennent pas d'étiquettes.

1.1 Les composantes du Machine Learning

On distingue quatre composantes essentielles au Machine Learning à savoir : les données, la tâche, les algorithmes et l'évaluation de performance jouant chacun un rôle spécifique [69].

- Les données : communément appelées *datasets*, les données sont les enregistrements que le Machine Learning a besoin pour faire des entraînements et construire un modèle de prédiction représentatif de l'ensemble des données du problème étudié. Les données sont sensées être fiables et cohérentes.
- La tâche à réaliser : la tâche à réaliser est la modélisation du problème qu'on souhaite solutionner.
- Les algorithmes : les algorithmes sont des « boites noires » configurables permettant à la machine de représenter les données fournies afin de créer un modèle représentatif des données. Ils sont aussi appelés "agents" et sont au cœur du mécanisme d'apprentissage automatique.
- La mesure de la performance : on entend par performance, la capacité d'apprentissage du Machine Learning sur une donnée **D** avec un algorithme **A** pour fournir un modèle

M. Cette performance **P** permet continuellement de valider le modèle **M**. C'est grâce aux mesures de performance que le Machine Learning évalue sa capacité d'apprentissage.

1.2 Les étapes d'application du Machine Learning face à un problème

L'application du Machine Learning face à un problème comporte souvent 4 étapes suivantes [63] :

- La récupération et la préparation des données ou processing (prétraitement) : cette étape permet de chercher les données qui représentent le problème à étudier et les filtrer (nettoyer) avant l'application d'un algorithme du Machine Learning, en supprimant les données non-cohérentes telles que les espaces.
- Le choix de l'algorithme : une fois que le problème a été bien ciblé (déterminer le type de problème à résoudre), on doit choisir un algorithme permettant de créer un modèle afin de résoudre le problème. Cet algorithme peut être choisi parmi les familles d'algorithmes supervisés, non-supervisés.
- La construction du modèle : une fois que le type d'algorithme est choisi, on construit le modèle en utilisant les données précédemment traitées.
- La phase de test du modèle et évaluation : après la conception du modèle, il faut le tester afin de déterminer sa performance en utilisant les outils de mesure de performance. Si le résultat du test n'est pas satisfaisant, il faudra déterminer la cause du problème qui réside soit du choix de l'algorithme soit dans la préparation de données. On peut également modifier les paramètres de l'algorithme pour gagner en performance.

Chapitre 2

Les modes d'apprentissage

Il existe plusieurs méthodes d'apprentissages automatiques qui peuvent être catégorisées en deux grands groupes : **supervisé**, **non-supervisé** comme le montre la figure 2.1. La différence majeure entre ces 2 groupes est que dans la méthode supervisée, les données sont dotées de labels alors que dans la méthode non-supervisée les données ne sont pas dotées de labels.

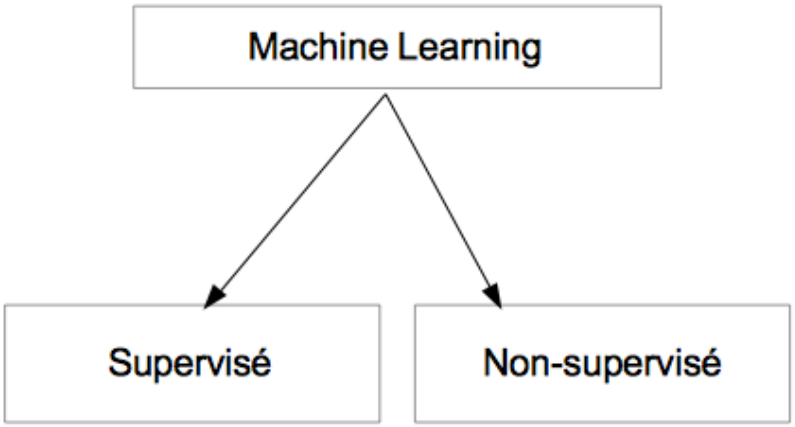

FIGURE 2.1 – Les modes d'apprentissage du Machine Learning.

2.1 Mode supervisé

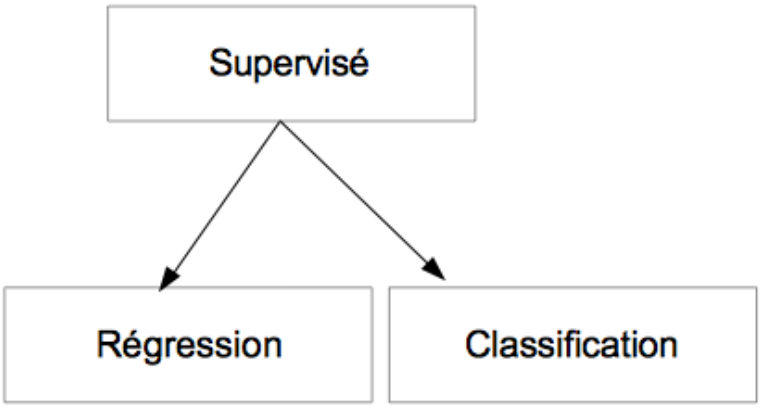

FIGURE 2.2 – Les types d'apprentissage du mode supervisé.

L'apprentissage supervisé est le processus d'apprentissage d'un algorithme à partir d'un ensemble de données dotés de label. Chaque enregistrement appartient à une classe bien déterminée. Les labels représentent les différentes classes dans lesquelles sont classées les données.
Les données contiennent des caractéristiques. Une caractéristique est une colonne de données, elle sert à décrire les données. L'ensemble de données noté x est représenté de la manière suivante :

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} \tag{2.1}$$

Avec n le nombre total d'enregistrements. Chaque occurence x_i possède une ou plusieurs caractéristique (s) notée (s) de la manière suivante :

$$x_i = \{x_i^1, x_i^2, .., x_i^d\}$$

où x_i correspond au i^e enregistrement dans les données et d le nombre total de caractéristiques.

La représentation complète du x :

$$x = \begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & \dots & x_1^d \\ x_2^1 & x_2^2 & x_2^3 & \dots & x_2^d \\ x_3^1 & x_3^2 & x_3^3 & \dots & x_3^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n^1 & x_n^2 & x_n^3 & \dots & x_n^d \end{bmatrix}$$

Les données x jouent un rôle essentiel dans le cadre de l'apprentissage. Elles permettent aux algorithmes d'apprendre et de construire un modèle permettant de représenter les données dans le cadre de l'apprentissage supervisé.
Les données sont obtenues soit par des simulateurs, soit par des systèmes réels. Il existe également de données en libre d'accès sur internet. L'accroissement technologique a permis d'augmenter la masse de données. En effet, chaque machine génère des informations de toutes sortes qui seront groupées et analysées. L'une des contraintes principales qu'exige le Machine Learning est la cohérence des données et qu'il n'y a pas dépendance entre les caractéristiques.
L'ensemble de données de formation comprend les données d'entrées x et les valeurs de classes y. L'algorithme d'apprentissage supervisé cherche à construire un modèle qui peut faire des valeurs de prédictions sur un ensemble de données de test. L'utilisation d'ensembles de données de formation plus vaste permet souvent de proposer des modèles avec un pouvoir prédictif plus élevé qui peuvent bien se généraliser sur de nouveaux ensembles de données.
Le résultat de la prédiction noté y dans le cadre de l'apprentissage supervisé s'obtient de la manière suivante :

$$y = h(x)$$

Avec $h(x)$ la fonction qui fait correspondre la donnée d'entrée x afin de prédire la sortie (y).

Les algorithmes appartenant à cette famille d'apprentissage font itérativement des prédictions sur x. L'apprentissage s'arrête lorsque l'algorithme atteint un niveau de performance jugé acceptable par le superviseur via les mesures de performances.
Il existe deux types de problèmes dans les méthodes supervisées à savoir : les problèmes de type classification et de type régression illustrés par la figure 2.2.

2.1.1 Classification

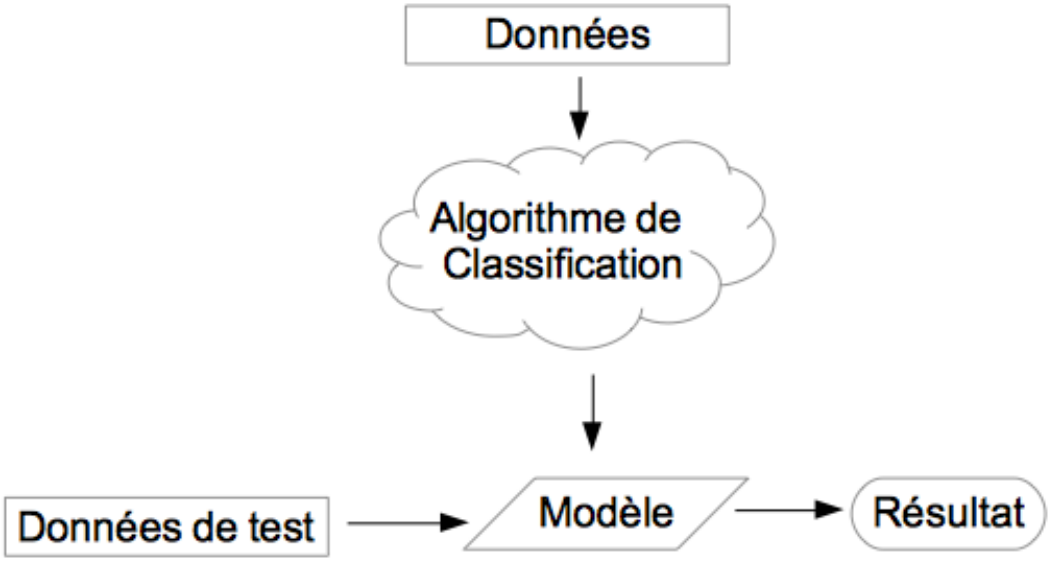

FIGURE 2.3 – Les étapes de la construction et de test du modèle.

On peut déterminer trois étapes dans le processus de classification. La première étape de la classification est la construction du modèle sur base d'un ensemble d'observations fournit dans le but d'établir l'existence de classes ou de clusters dans les données (Figure 2.3 sur l'axe verticale).
La deuxième étape consiste à utiliser le modèle précédemment construit afin de classifier les nouvelles données à tester (figure 2.3 sur l'axe horizontale). La dernière étape est l'évaluation du modèle selon les critères non-exhaustifs tels que : la robustesse, la précision, l'exactitude (Accuracy), etc.
Dans la classification, le but est de construire une droite appelée **decision boundary** afin de séparer les classes. Les données se trouvant de part et d'autre de la droite doivent appartenir à une même classe. Il existe plusieurs algorithmes permettant de résoudre les problèmes liées à la classification illustrés par la figure 2.4.

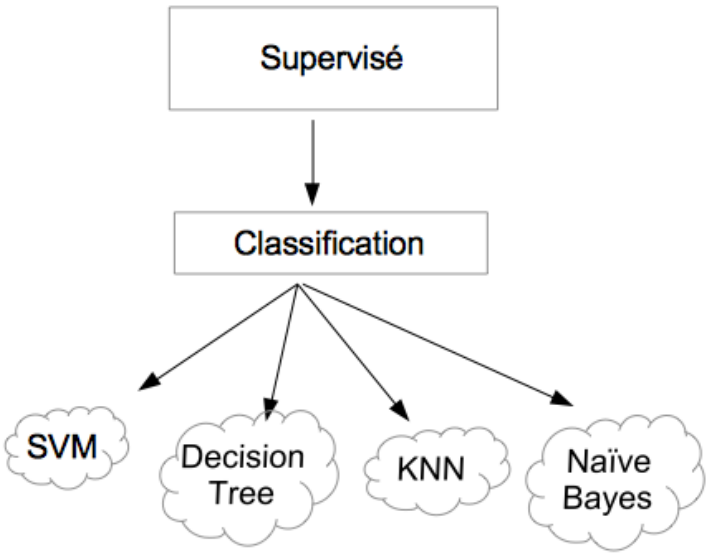

FIGURE 2.4 – Les algorithmes de classification.

2.1.1.1 KNN

Le **K**-**N**earest **N**eighbors (KNN) est un algorithme permettant de classer les échantillons sur base de leur similarité. La règle de fonctionnement du KNN est décrite de la manière suivante :
Sur base d'un ensemble de données **D** et d'une fonction de mesure de distance **M** et un nombre entier **K**, l'algorithme va chercher dans **D** les **K** points les plus proches de x_i en terme de mesure de distance selon la mesure **M**.
A titre de rappel, les différentes mesures de distances sont Manhattan, Hamming, Minkowski et Euclidienne. Chaque instance vote pour leur classe et la classe qui a le plus de votes est prise comme la valeur de la prédiction. Il est judicieux de choisir une valeur K avec un nombre impair pour éviter une égalité, à l'inverse, il faut utiliser un nombre pair pour K lorsqu'il y'a un nombre impair de classes.
Le KNN est un algorithme lent car toutes les instances des données sont revues à chaque fois. Un autre point faible du KNN est qu'il est peu robuste car il est sensible aux caractéristiques non-pertinentes ainsi qu'aux caractéristiques corrélées.

2.1.1.2 SVM

Le SVM est un algorithme d'apprentissage automatique de type supervisé qui est utilisé pour les problèmes de classification. Il dispose de sa méthode pour trouver la frontière entre les classes. Cette frontière est appelée **l'hyperplan**. L'hyperplan doit être placé à une grande

distance des **Supports Vectors**. Les Supports Vectors sont des points issus des données d'entraînement x les plus proches de l'hyperplan. Il faut maximiser la distance entre les Supports Vectors et l'hyperplan. L'idée de cette maximisation est d'anticiper les erreurs de classification. En d'autre terme, cela garantie qu'un nouvel exemple se situant entre l'hyperplan et un Support Vector, sera de la même classe que le Support Vector. La marge est la distance minimale de l'hyperplan à un des points d'entraînement.

Dans certains cas il est impossible de trouver une ligne droite permettant de séparer les classes, il faut alors trouver une transformation capable de classer les éléments. Cette technique s'appelle **Kernel trick** ou fonctions noyaux. Les fonctions noyaux sont des fonctions qui prennent un espace d'entrée de dimension faible et le transforme dans un espace de dimension plus élevée, c'est-à-dire qu'elles convertissent le problème non séparable en problème séparable.
L'hyperplan est calculé via l'hypothèse $h(x)$ de la manière suivante :

$$h(x) = w_1.x_1 + w_2.x_2 + w_3.x_3 + ... + w_n.x_n + b$$

ou de façon plus simplifiée :

$$h(x) = \sum_{i=1}^{n} w_i.x_i + b$$

Avec w_i le poids de x_i et b le biais. Le but est de déterminer le poids w_i. L'interprétation de l'hypothèse $h(x)$ pour un point x_i est la suivante :

- Si $h(x_i)$ >= 0 alors x_i appartient à la première classe.
- Sinon x_i appartient à la seconde classe.

On doit trouver les meilleurs paramètres w_i et b pour l'hyperplan afin de maximiser la marge. Le SVM est très efficace lorsqu'on ne dispose pas assez de données d'entraînement. A l'inverse, lorsqu'on dispose de beaucoup de données le SVM a tendance à baisser en performance. L'hyper-paramètre **C** régit la performance du SVM. Cet hyper-paramètre sert à fixer le compromis entre la minimisation de l'erreur d'apprentissage et la maximisation de la marge. En pratique, le comportement du SVM est sensible à la valeur de C uniquement si les données d'apprentissage ne sont pas séparables. Dans ce cas, il existe des valeurs critiques qui peuvent compromettre la performance du classifieur. Une très grande valeur de C peut faire que la fonction objective minimisée par le SVM ne soit plus convexe et empêcherait sa convergence. Une très faible valeur de C tend à diminuer la capacité du classifieur.

2.1.1.3 Naïve Bayes

Le Naïve Bayes [97] est une technique de classification basée sur le théorème de Bayes avec une hypothèse d'indépendance dans les caractéristiques. Autrement dit, un classificateur Naïve Bayes suppose que la présence d'une caractéristique dans une classe n'est pas liée à la présence d'une autre caractéristique. Ainsi, cette hypothèse d'indépendance épargne le calcul de la covariance. Le champ d'action du Naïve Bayes est très diversifié dont notamment la classification de textes qui se base sur la fréquence d'apparition des mots. Le théorème de Bayes se décrit de la manière suivante :

$$P(y \mid x) = \frac{P(x \mid y).P(y)}{P(x)}$$

Le théorème précédent se traduit de la manière suivante [84] :

$$P(y \mid x_1, x_2, .., x_n) = \frac{P(x_1, x_2, .., x_n \mid y).P(y)}{P(x_1, x_2, .., x_n)}$$

Et simplifié par :

$$P(y \mid x_1, x_2, .., x_n) = \frac{P(y).\prod_{k=1}^{n} P(x_k \mid y)}{P(x_1, x_2, .., x_n)}$$

La classe prédite y est calculée de la manière suivante :

$$y = \arg\max_{y} P(y) \prod_{k=1}^{n} P(x_k \mid y)$$

Via cette formule, il faut déterminer quelle classe a une plus grande probabilité d'apparition.

- $P(y \mid x)$ est la probabilité **a posteriori** de la classe ciblée (y) sous la condition de x.
- $P(x \mid y)$ la vraissemblance (**likelihood**) de probabilité de l'événement x sachant y.
- $P(y)$ et $P(x)$ respectivement la probabilité **a priori** de la classe ciblée de y et la probabilité **a priori** de x.

Il existe plusieurs extensions dont **Gaussian Naive Bayes** ou **la distribution normale**.

2.1.1.4 Arbre de décision

L'arbre de décision est une méthode de prédiction puissante et très populaire utilisée dans le cadre de l'apprentissage automatique. Les arbres de décision sont des règles de classification qui basent leur décision sur une suite de tests associées aux caractéristiques organisées de manière arborescentes. Les valeurs des caractéristiques peuvent être binaires, n-aires. L'idée

centrale d'un arbre de décision est de diviser de manière récursive et efficacement les échantillons de l'ensemble d'apprentissage par des tests définis à l'aide des caractéristiques jusqu'à obtenir un sous-ensemble homogène (c'est-à-dire appartenant à la même classe). Le premier sommet est appelé la racine de l'arbre. Un arbre est composé de nœuds et chaque nœud est affecté d'une proportion qui permet de voir son appartenance à une classe. Les nœuds internes sont appelés nœuds de décision. Chaque nœud de décision est soumis par un test sur la branche au dessus de ce nœud. Les deux opérateurs relatifs aux arbres sont :

- Décider si un nœud est terminal et lui affecter une classe : la décision d'un nœud terminal est lorsque tous les exemples associés à ce nœud, ou du moins la plupart d'entre eux sont dans la même classe. Un nœud est dit aussi terminal s'il n'y a plus de caractéristiques non utilisées dans la branche correspondante.
- Si un nœud n'est pas terminal, on sélectionne un test à lui attribuer : chaque nœud intermédiaire réalise un test portant sur une variable dont le résultat indique la branche à suivre dans l'arbre. L'objectif est de créer un modèle qui prédit la valeur d'une variable cible en apprenant des règles de décision simples déduites des caractéristiques de données.

La sélection de caractéristiques dans un arbre de décision est faite selon le critère de gain d'information et de l'entropie. L'entropie est la mesure de l'incertitude dans les données [97]. En d'autres termes, l'entropie mesure l'homogénéité de l'ensemble. Elle représente également le désordre au sein de l'ensemble de données. Elle donne de l'information en terme de label de la classe. Pour trouver la caractéristique à tester, il faut d'abord calculer l'entropie de l'ensemble des classes initiales possibles de la manière suivante [11] :

$$Entropie(S) = \sum_{i \in classes} (-p_i) \log_2(p_i)$$

avec p_i la probabilité d'obtenir la classe *i* et S représente l'ensemble de données considérées. La valeur de l'entropie (S) se situe entre 0 et 1. Une entropie nulle (0) signifie que le nœud est pure. Par contre si une entropie est à 1, les classes ont la même probabilité d'apparition. Il faut ensuite déterminer quelle caractéristique à tester en premier lieu, puis en deuxième lieu, ainsi de suite. Pour cela, il faut calculer le gain d'entropie.

$$Gain(S, C) = Entropie(S) \; - \sum_{c \in Caractristiques\ (C)} \frac{\mid S_c \mid}{\mid S \mid} * Entropie(S_c)$$

C est l'ensemble des caractéristiques, $\mid S_c \mid$ est le nombre d'occurrence correspondant à la classe c. $\mid S \mid$ est la cardinalité de l'ensemble S. Les valeurs de y de prédiction d'un arbre

de décision peuvent être *continues*, dans ce cas, on parle d'**arbre de régression**. Il reparti les individus en groupe homogène par rapport à la variables à prédire en tenant compte d'une hiérarchie de la capacité prédictive des variables considérées. Cette hiérarchie permet de visualiser les résultats dans un arbre. Les principaux algorithmes d'apprentissage basés sur les arbres de décisions sont J48, CHAID CART, C4.5 et ID3.

L'algorithme ID3 utilise une recherche gourmande. Il sélectionne un test en utilisant le critère de gain d'information, et n'explore jamais la possibilité de choix alternatifs. Au sein de cet algorithme, une seule caractéristique à la fois est testée pour prendre une décision et ne gère pas les caractéristiques numériques et les valeurs manquantes. Tant disque l'algorithme CHAID CART peut gérer à la fois les variables numériques et catégorielles et peut facilement gérer les valeurs aberrantes. Le **C4.5** est une version améliorée du ID3. Contrairement à l'algorithme ID3, le C4.5 accepte à la fois les caractéristiques de type continues et discrètes. Il utilise le critère d'entropie pour sélectionner les nœuds tant disque l'algorithme **CHAID CART** utilise le critère de GINI pour sélectionner les nœuds. [97] le calcul du GINI s'effectue de la manière suivante :

$$GINI = \sum (p_i).(1 - p_i)$$

Avec avec p_i la probabilité d'obtenir la classe i et 1-p_i le complémentaire de la probabilité d'obtenir la classe i. Les valeurs du GINI s'interprètent de la même manière que celles de l'entropie.

2.1.2 La régression

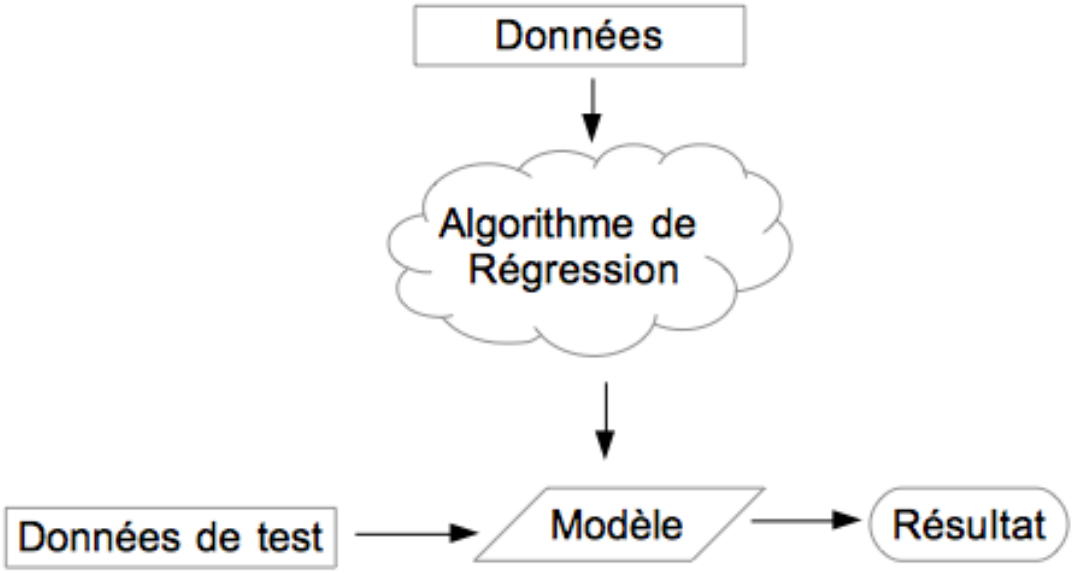

FIGURE 2.5 – Les étapes de la construction et de test du modèle.

Par rapport aux problèmes de type classification qui produisent des résultats qualitatifs, les problèmes de type régression prédisent des résultats de type continus sur base des données observées. La régression doit estimer une fonction qui fait correspondre les données en entrée avec les données en sortie. Elle est utilisée pour prédire sur base des données représentatives de chaque situation par exemple pour prédire la consommation d'énergie, le prix des maisons, estimer le prix du stock, etc [93].
Il existe plusieurs types de régressions dont nous allons en examiner quelques uns.

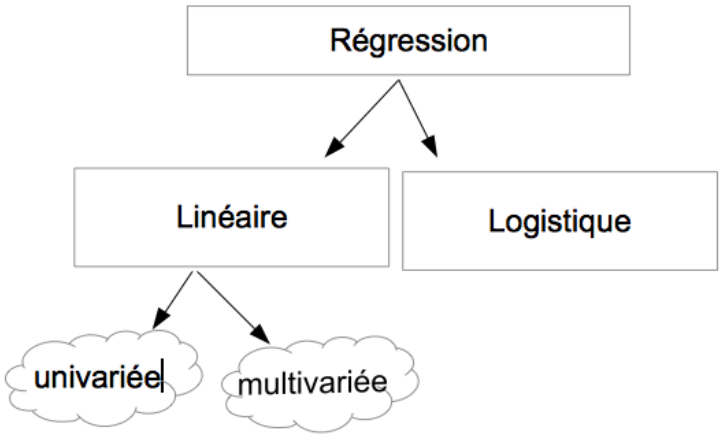

FIGURE 2.6 – Les différents types de régression.

La **régression linéaire** est un modèle linéaire qui suppose une relation linéaire entre les variables d'entrées et la variable de sortie unique.
La régression linéaire fait l' hypothèse que les données proviennent d'un phénomène qui a la forme d'une droite, c'est-à-dire qu'il existe une relation linéaire entre l'entrée (les observations) et la sortie [103]. Un problème de **régression simple** (ou univariée) où le modèle dépend d'une caractéristique unique x^1. Le modèle est défini de la manière suivante :

$$h_\theta(x) = \theta^0 + \theta^1 x^1$$

En outre quand il y a plusieurs caractéristiques d'entrée, on se réfère souvent à la méthode de **régression linéaire multiple** ou la régression linéaire multivariée qui est une généralisation du cas univarié. Le but est de déterminer le poids des paramètres θ_0 et θ_1 dans le cas de la régression linéaire simple. Autrement dit, il faut choisir θ^0 et θ^1 telle que l'hypothèse $h_\theta(x)$ soit aussi proche de Y (la vraie valeur).
Cette détermination passe par le choix de la **Cost function** (la fonction de coût). La Cost function permet d'adapter la meilleure droite possible aux données. Elle est définie formellement de la manière suivante :

$$J(\theta) = \frac{1}{2N} \sum_{i=1}^{N} (h_\theta(x)^i - y^{(i)})^2$$

Avec

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_n \end{bmatrix} \tag{2.2}$$

Il faut trouver les valeurs du θ afin de minimiser l'erreur de prédiction. Cette minimisation se fait grâce au **Gradient Descent**. Le Gradient Descent est un algorithme d'optimisation permettant d'optimiser les valeurs des coefficients θ et minimisant par itération l'erreur du modèle sur les données d'entraînement. Le Gradient Descent commence par des valeurs aléatoires pour chaque coefficient. **La somme des carrés des erreurs** est calculée pour chaque paire de valeurs d'entrée et de sortie. Un taux d'apprentissage est utilisé en tant que facteur d'échelle et les coefficients sont mis à jour dans l'optique de minimiser l'erreur. Le processus est répété jusqu'à ce qu'une somme minimum de l'erreur quadratique est atteinte ou qu'il n'y ait pas d'amélioration supplémentaire possible. Le fonctionnement du Gradient Descent est le suivant :

- On initialise aléatoirement les différentes valeurs du θ.
- On continu à changer les valeurs du θ afin de réduire $J(\theta)$ pour obtenir une valeur minimum.

Plus formellement, la mise à jour des coefficients est illustrée par l'algorithme suivant :
Répéter jusqu'à la convergence {

θ_j := θ_j - α $\frac{\partial}{\partial \theta_j}$ θ $J(\theta)$

}

α $\frac{\partial}{\partial \theta_j}$ θ $J(\theta)$ est la dérivée partielle par rapport à θ_j.

α est le taux d'apprentissage (learning rate) afin de contrôler la taille de la descente. Il est question du bon choix du paramètre α. En effet, si α est trop faible, la convergence sera très lente, à l'inverse si α est très grande, la convergence sera très rapide et risque de rater le point optimale.
On peut également minimiser la Cost function via la Normal Equation. La Normal Equation

est obtenue de la manière suivante :

$$\frac{\partial}{\partial \theta_j} \theta J(\theta) = \frac{x^T(x\theta - y)}{N}$$

On peut comparer les deux méthodes de minimisation de la manière suivante :

- En opposition au Gradient Descent, la Normal Equation n'exige pas la détermination du α.
- Le Gradient Descent fait plusieurs itérations pour converger alors que la Normal Equation ne fait pas d'itération.
- Le Gradient Descent s'améliore avec une grande quantité de données d'entraînement.
- Le Gradient Descent est un algorithme beaucoup plus général d'optimisation par rapport à la Normal Equation.

Il existe aussi d'autres méthodes de régression dans la catégorie multivariée telle que la **régression polynomiale** qui introduit la non-linéarité dans un modèle pourtant linéaire. La régression linéaire polynomiale hérite des principes de la régression linéaire multivariée, elle sert à relier les caractéristiques par un polynôme de degré d.

La régression logistique est une technique supervisée de type régression visant à décrire la relation pouvant exister entre une variable à expliquer et une ou plusieurs variable explicatives. Quand la variable à expliquer contient deux modalités (dont les valeurs seraient VRAI/FAUX, $0/1$, ou OUI/NON) alors on utilise la régression binaire. A l'inverse lorsque la variable à expliquer contient plus de deux modalités et sans une relation d'ordre alors on utilise la régression logistique polychotomique nominale. Lorsqu'il y a plus de deux modalités et avec relation d'ordre on utilise alors la régression logistique polychotomique ordinale. Pour toutes les méthodes de régression pré-citées, le but est de modéliser une ou plusieurs probabilités concernant l'appartenance aux classes en fonction des caractéristiques explicatives pouvant être qualitatives ou quantitatives.
Le but de la régression logistique binaire est de modéliser la probabilité d'appartenance de x à l'une des deux modalités. Ainsi la probabilité que la variable à expliquer y appartient à la première modalité sachant qu'on a x est notée de la manière suivante :

$$P(y = 1 \mid x)$$

En reprenant le modèle de régression linéaire, on pourrait être tenté d'exprimer les probabilités $P(y = 1)$ et $P(y = 0)$ sous la forme d'une combinaison linéaire sous la forme : $\theta^0 + \theta^1 x^1$ et de tester les coefficients de régression (θ^0, θ^1).
Néanmoins une telle équation ne garantit pas que les valeurs prédites de $P(y = 1)$ et $P(y = 0)$ soient comprises entre 0 et 1 pour toutes les combinaisons de valeurs prises par les caractéristiques explicatives.
Afin de remédier à ce problème, il faut appliquer une transformation sur $P(y = 1)$. Il s'agit de la fonction de lien ou Logit. La fonction de Logit est définie de la manière suivante :

$$Logit(P(y = 1)) = \log\left(\frac{P(y = 1)}{1 - P(y = 1)}\right)$$

Comme dans le cadre de la régression linéaire, il faut estimer les paramètres θ^0 et θ^1. La probabilité $P(y = 1)$ est calculée de la manière suivante :

$$P(y = 1) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-(\theta^0 + \theta^1 x^1)}}$$

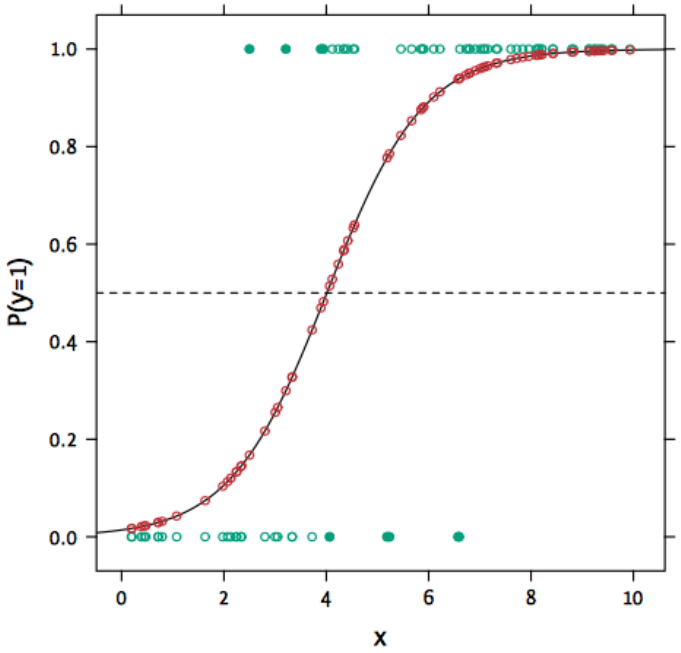

FIGURE 2.7 – Exemple d'une fonction Sigmoïde [36].

Il est important d'évaluer le facteur côte chance appelé l'Odds, c'est-à-dire la probabilité de succès par rapport à l'échec. Plus formellement :

$$Odds = \frac{P(y=1)}{1-P(y=1)}$$

Si la probabilité de succès $P(y=1)$ est inférieure à la probabilité de l'échec $P(y=0)$ alors l'Odds est inférieure à l'unité.
Le rapport des chances est exprimé par l'Odds Ratio (OR). Il s'obtient via la formule suivante :

$$OR = \frac{P(y=1)/1-P(y=1)}{P(y=0)/1-P(y=0)}$$

L'interprétation des résultats d'une régression logistique est très proche de l'interprétation des résultats d'une régression linéaire.

2.1.3 Réseau de neurones

Le réseau de neurones est une technique de calcul qui se base sur le fonctionnement identique à celui des neurones biologiques. Un réseau de neurones artificiels est constitué de neurones artificiels qui sont reliés par couches successives. Chaque neurone possède des entrées pondérées et une sortie qui est reliée à tous les neurones de la couche suivante. Le neurone calcule la somme de ses entrées puis cette valeur est passée à la fonction d'activation pour produire sa sortie. Cette somme est calculée de la manière suivante :

$$\sum_{i=1}^{d} x_i.w_i + w_0$$

On distingue plusieurs types de réseaux de neurones tels que le perceptron simple et le perceptron multi-couches.

Le perceptron simple est un modèle de réseau de neurones à une seule couche. Les entrées x^1,x^2,..,x^d correspondent aux caractéristiques et w^1,w^2,..,w^d sont les poids associés aux caractéristiques ainsi que θ le seuil (biais).

On calcule la sortie via la fonction d'activation. Il existe plusieurs types de fonctions d'activations à savoir : ReLU, Sigmoïde, Gaussienne, Identité, Linéaire, Seuil, Pas unitaire, etc. Pour les problèmes de classification binaire, on peut utiliser une fonction de seuil de la manière suivante :

$$O = \begin{cases} 0, & \sum_{i=1}^{d} x_i.w_i + w_0 < 0. \\ 1, & \text{Ailleurs.} \end{cases} \tag{2.3}$$

On peut également utiliser la fonction Sigmoïde (vue précédemment) pour prédire la probabilité d'appartenir à la classe positive.

Le perceptron multi-couches est un réseau de neurones composé de plusieurs couches dont des couches cachées. Les couches cachées sont des couches situées entre la couche d'entrée et la couche de sortie. Elles permettent d'augmenter la puissance de prédiction du réseau. Chaque couche est constituée d'un nombre variable de neurones, les neurones de la couche de sortie correspondant toujours aux sorties du système. Lorsqu'une information circule uniquement de la couche d'entrée vers la couche de sortie il s'agit alors d'un réseau de type **feedforward**. Un autre mode de propagation d'information est le **feedforward backpropagation** ou la rétro-propagation permet la circulation d'une information de la couche d'entrée vers la couche de sortie, puis de la couche de sortie vers la couche d'entrée.
Au sein du perceptron multi-couches de type feedforward, les neurones d'une couche sont reliés à la totalité des neurones des couches suivantes. Ces liaisons sont soumises à un poids altérant l'effet de l'information sur le neurone de destination. Le poids de chacune de ces liaisons est très indispensable au fonctionnement du réseau : la mise en place d'un perceptron multi-couche à rétro-propagation permet de faire par la détermination des meilleurs poids applicables à chacune des connexions entre les neurones.
Le réseau de neurones de type feedforward rétro-propagation hérite de toutes les caractéristiques du réseau de neurones feedward. La différence est qu'au sein du réseau de neurones de type feedforward backpropagation les poids sont mises à jour après le calcul de l'erreur de sortie. Les étapes importantes de l'exécution de la rétro-propagation sont :

- La propagation de l'entrée jusqu'à la sortie.
- Le calcul de l'erreur de sortie.
- La rétro-propagation de l' erreur jusqu'aux entrées (c'est-à-dire la mise à jour des poids).

La mise à jour des poids (w_i) se font dans une itération appliquant la formule suivante :

$$w_i = w_i + \epsilon.(y_i - o).x_i$$

avec y_i la prédiction de x_i et ϵ correspond à la valeur du pas d'apprentissage défini par l'utilisateur. Une grande valeur du pas (ϵ) d'apprentissage augmentera la probabilité de sauter la valeur optimale. Un pas trop petit nécessitera un grand nombre d'itérations pour trouver la valeur optimale.
La mise à jour des poids s'arrête lorsqu'on a atteint le nombre maximum d'itération ou qu'il y a convergence des poids.

Les réseaux de neurones sont utilisés dans les réalisations de plusieurs tâches telles que : la reconnaissance des formes (des chiffres, visage, etc.). Les réseaux de neurones ont également montré leur efficacité dans le cadre de la biométrie.

2.2 Non-supervisé

Il existe deux types de clustering à savoir le **clustering hiérarchique** [80] et le **clustering non-hiérarchique** qui respectent deux principes à savoir : les individus d'un même groupe doivent se ressembler et les individus de groupes différents ne doivent pas avoir de ressemblance.
Le but du clustering hiérarchique est de réaliser une suite de clusters (groupes) inclus les uns dans les autres. Ces groupes sont liés les uns aux autres par des relations de différents niveaux et peuvent êtres représentés par une hiérarchisation arborescente. On utilise cette technique lorsqu'on ignore le nombre n de clusters dans lesquels on va regrouper les observations.
Le clustering hiérarchique considère au démarrage que tous les éléments sont des classes. Pour tous les éléments, il évalue les distances entres les points et les représentants des classes. Ensuite on fusionne les deux classes les plus proches pour obtenir le nombre de classes souhaitées. Au sein de cette hiérarchisation arborescente, on distingue deux types de familles d'algorithmes :

- La famille des algorithmes ascendants ou agglomératifs crée des agglomérations de manière successive par groupes de 2 objets.
- A l'inverse, la famille des algorithmes descendants réalisent des dichotomies de manière progressive sur l'ensemble des observations.

L'utilisation de ces algorithmes amène à déterminer les critères sur lesquels on se base pour déterminer que deux éléments sont de la même famille, la notion de distance, ou bien combien de classes (groupes). En ce qui concerne le critère de la distance, il est question de définir une mesure de similarité entre deux éléments, c'est-à-dire calculer la distance qui sépare deux éléments afin de décider s'ils appartiennent à un même sous-ensemble. On peut utiliser la technique d'Euclide pour calculer la distance entre deux points A et B de cette manière :

$$d(A,B) = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2} \tag{2.4}$$

Dans certaines situations, on utilise la distance de Manhattan :

$$d(A,B) = \mid x_B - x_A \mid + \mid y_B - y_A \mid \tag{2.5}$$

Rappelons que chaque élément est caractérisé par son repère cartésien (avec x pour l'axe des abscisses et y pour l'axe des ordonnées). Il faut donc que la distance entre les éléments appartenant à une même classe soit la plus petite que possible et la distance entre les éléments appartenant à deux classes distantes soit maximale. Le critère d'agrégation permet de se poser la question de savoir comment calculer la distance en présence de groupes ? trois critères permettent de répondre à ces questions :

- Le critère du lien maximum qui détermine deux éléments éloignés.
- Le critère du lien minimum qui détermine deux éléments les plus proches.
- Le critère du lien moyen utilise la moyenne des distances entre les éléments de chaque cluster pour effectuer les regroupements.

Le clustering non-hiérarchique a le même objectif que le clustering hiérarchique à savoir la détermination du nombre *n* de classe à constituer. Le clustering non-hiérarchique énumère toutes les possibilités de regroupement et conserve la meilleure.

2.2.1 K-Mean

Les algorithmes basés sur le clustering non-hiérarchique sont nombreux mais l'un des plus connus reste **K-Mean** dont le principe de l'algorithme est le suivant :

1. Il faut fixer *K*, le nombre de classe que l'on souhaite avoir.
2. On choisit de façon aléatoire *c* point pour représenter les centres de cluster.
3. Puis pour chaque élément, on calcule la distance entre chaque point de données et les centres de cluster.
4. Ensuite, on affecte le point de données au centre de cluster dont la distance par rapport au centre de cluster est au minimum de tous les centres de clusters.
5. On recalcule le nouveau centre de cluster, on recalcule également la distance entre chaque point de données et les nouveaux centres de grappe obtenus précédemment.
6. Ensuite, on affecte à cet élément la classe dont sa distance est minimale. Si aucun point de données n'a été modifié, le processus s'achève, au cas échéant on refait le calcul à partir de l'étape 4.

2.2.2 DBSCAN

Le DBSCAN est un algorithme non-supervisé très populaire basé sur la notion de densité utilisé pour des analyses prédictives. Il est un algorithme de clustering considéré comme une alternative au K-Mean. Au sein de cet algorithme, il n'est pas nécessaire de spécifier le nombre de clusters car l'algorithme le découvrira par lui même à travers les données. Ce qui constitue un avantage que DBSCAN a sur le K-Mean car l'utilisation du K-Mean implique l'initialisation nombre de clusters pendant la phase de démarrage. Le DBSCAN a deux paramètres à savoir : ϵ (*epsilon*) et *MinPts*. Un point est un voisin d'un autre point s'il est situé à une distance fixée appelée ϵ (*epsilon*).
La notion de densité se définit de la manière suivante : un point est dit dense si le nombre de ses voisins dépasse un seuil fixé appelé *MinPts*. Le but du DBSCAN est d'identifier les régions denses qui sont mesurées avec les deux paramètres. Toutes les cordonnées des points avec un nombre de voisins plus grand ou égal aux MinPts sont marquées comme des points intérieurs. A l'inverse, si le nombre de voisins d'un point est inférieur au MinPts alors ce point est appelé un **border point**.

Chapitre 3

Évaluation

Afin de déterminer la qualité du modèle construit dans le cadre de l'apprentissage, il faut évaluer la performance du modèle.
L'évaluation permet de savoir si le modèle a atteint un bon niveau de prédiction et évaluer les erreurs de prédictions. Le modèle doit être évalué sur un ensemble de données statistiquement indépendant de celui sur lequel il a été formé. Une évaluation plus juste mesure la performance du modèle sur les données qui n'a pas encore été utilisée dans la phase de la création du modèle. Cela donne une estimation de l'erreur de généralisation, qui mesure à quel point le modèle se généralise sur des nouvelles données. Il existe plusieurs indicateurs permettant d'évaluer la performance d'un modèle, mais dans le cadre de la réalisation de ce mémoire nous verrons principalement les mesures les plus fréquemment utilisées.

3.1 La validation croisée

Une façon de générer de nouvelles données de test consiste à contenir une partie du jeu de formation et à l'utiliser uniquement pour l'évaluation. Cette méthode est connue sous le nom de validation de **hold-out**. La méthode plus générale est connue sous le nom de **K-fold cross-validation** (validation croisée). La validation croisée est une technique pour évaluer les modèles prédictifs en divisant l'échantillon d'origine dans un ensemble de formation pour tester le modèle [83]. Il existe au moins trois méthodes de validation croisée : *Testset validation*, *K-fold cross-validation [87]* et *Leave-one-out-cross-validation (LOOCV)*.

Dans la méthode du Testset validation, on divise l'échantillon de taille n en deux sous échantillons, le premier d'apprentissage (souvent supérieur à 60 % de l'échantillon) et le second de test. Le modèle est bâti sur l'échantillon d'apprentissage et validé sur l'échantillon de test. Dans la validation croisée K-fold cross-validation, l'échantillon d'origine est divisé en

sous-échantillons de façon aléatoire k de taille égale. Parmi les sous-échantillons k, un sous-échantillon est retenu comme les données de validation pour tester le modèle, et les autres sous-échantillons k-1 sont utilisés comme données de formation. Le processus de la validation croisée est répété k-fois, avec chacun des sous-échantillons de k utilisé exactement une fois que les données de validation. L'avantage de cette méthode est que toutes les observations sont utilisées pour la formation et la validation. Chaque observation est utilisée pour la validation exactement une fois.

Le **leave-one-out-cross-validation** est une validation croisée de K-fold prise à son extrémité, avec k égal à N, le nombre de points de données dans l'ensemble. Cela signifie que N fois séparée, l'approximateur de fonction est formé sur toutes les données sauf pour un point et une prédiction est faite pour ce point.

3.2 Matrice de confusion

		Predicted class	
Actual Class		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

FIGURE 3.1 – La matrice de confusion extrait du [77] .

La matrice de confusion est un outil d'évaluation de la classification basé sur un tableau à deux dimensions (Figure 3.1). Elle est une matrice permettant d'évaluer l'intensité de la liaison entre des données de référence et le résultat fournit par le modèle [35] [41]. L'information des rangées (les données horizontales) correspond habituellement aux classes thématiques de la carte de référence et les colonnes verticales contiennent l'information thématique résultant de la classification, c'est-à-dire les résultats de la classification provenant du test du modèle. La matrice de confusion tient compte des quatre situations [81] : **V**rais **p**ositifs (VP), **V**rais **n**égatifs (VN), **F**aux **p**ositifs (FP), **F**aux **n**égatifs (FN). A noter que les FN et FP sont les deux types d'erreurs de classification qu'on souhaite minimiser.

- Le nombre des VP est le nombre de résultats jugé être positifs par le modèle ainsi que la référence.
- Le nombre des VN est le nombre de résultats jugés être négatifs par le modèle ainsi que par la référence.

- Le nombre des FP est le nombre de résultats jugés être positifs par le modèle mais la référence estime qu'ils sont en réalités négatifs.
- Le nombre des FN est le nombre de résultat jugés être négatifs par le modèle mais la référence estime qu'ils sont en réalités positifs.

Les situations précédemment citées permettent de calculer les indicateurs de performance suivant à savoir : la précision (P), le rappel (R), les taux d'erreurs (TE) et l'exactitude (Accuracy).

- Accuracy : l'accuracy (l'exactitude) calcule la proportion d'éléments bien classés. Par exemple, pour les situations vraies, l'accuracy s'obtient de la manière suivante :

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN}$$

- Le taux d'erreur (TE) : le taux d'erreur est le rapport entre le nombre d'éléments identifiés incorrectement (FP et FN). Ce rapport s'effectue de la manière suivante :

$$TE = \frac{FP + FN}{FP + VP + FN + VN}$$

- La précision : la précision mesure combien de prédictions positives qui sont réellement positives. Elle calcule l'exactitude des prédictions positives. On l'obtient de la manière suivante :

$$Precision = \frac{VP}{VP + FP}$$

- Rappel (R) : Le rappel indique à quel point le classifieur couvre les données. Le rappel indique le pourcentage d'éléments qu'il détecte par rapport à l'ensemble d'éléments détectables pour une classe donnée. Le rappel est aussi appelé comme étant la mesure de la sensibilité. Le rappel est connu également connu comme le taux des VP. Le rappel est donné par la formule suivante :

$$Rappel = \frac{VP}{VP + FN}$$

- Le taux des FP : Le taux des FP reflète la fréquence avec laquelle le modèle fait une erreur en classant les situations normales comme comme anormales. Ce taux est donné par la formule suivante :

$$Taux\,FP = \frac{FP}{FP + TN}$$

- A l'inverse, le taux des vrais négatifs s'obtient de la manière suivante :

$$SP = \frac{VN}{VN + FP}$$

  Elle est considérée comme étant la mesure de la spécificité (SP) car elle mesure la proportion de négatifs correctement identifiés.

- La mesure f_1-score : La mesure f_1 ou communément appelé f_1-score est la moyenne harmonique pondérée de précision et de rappel (R), elle s'obtient de la manière suivante :

$$f_1 - score = \frac{2.P.R}{P + R}$$

3.3 Courbe ROC (Received Operating Characteristic)

La courbe ROC est un outil de visualisation de la performance dans le cadre de l'apprentissage supervisé d'un classifieur. La courbe ROC montre la sensibilité du classificateur en traçant le taux de VP au taux de FP. En d'autres termes, elle montre combien de classifications positives correctes peuvent être obtenues lorsqu'on augmente de plus en plus de FP.

La courbe ROC n'est donc pas un seul point mais sur toute la courbe car elle fournit des détails nuancés sur le comportement du classificateur.
"L'aire sous la courbe" est aussi dénotée AUC pour « Area Under the ROC », permettant de comparer plusieurs modèles. L'aire sous la courbe est un indice calculé pour les courbes ROC. L'AUC correspond à la probabilité pour qu'un événement positif soit classé comme positif par le test sur l'étendue des valeurs seuil possibles. Un seuil t est une valeur telle que les éléments ayant un score notons s > t sont assignés à la classe 1, et sinon à la classe 0 dans le cadre de la classification binaire. Le score s représente le degré auquel une instance est membre d'une classe. Le score de classification indique la proportion d'exemples bien classés (dans les deux classes). La difficulté réside dans la valeur du seuil limite qui va décider si le test est déclaré positif ou négatif car, si on place le seuil trop bas, il y'aura beaucoup de valeurs positives, la sensibilité du test sera élevée, ainsi obtenir un grand nombre de FP et on risque d'inclure les positives à tort.

A l'inverse, un seuil trop élevé offrira au test une bonne spécificité (peu de valeurs positives), mais une mauvaise sensibilité car il laissera un grand nombre de vrais positifs qui n'entreront pas dans le processus de soins. La meilleure valeur du seuil (celle qui rend le test le plus efficace) est celle qui maximise en même temps la sensibilité et la spécificité du test, c'est-à-dire la valeur qui détectera un maximum des vrais positifs et écartera un maxi-

mum des faux positifs. Chaque valeur du seuil produit un point dans la ROC. Le point est représenté par le couple 1-spécificité en abscisse et sensibilité en ordonné [31].

3.4 Mean Squared Error (MSE)

La mesure de l'erreur quadratique moyenne nécessite une cible de prédiction ou d'estimation avec un prédicteur ou un estimateur que l'on dit être la fonction des données. La MSE est définie comme la moyenne des carrés des erreurs.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(h_\theta(x)^i - y^i)^2$$

Lorsque la valeur de MSE est nulle, cela signifie qu'il y a une précision parfaite trouvée.

3.5 Root Mean Squared Error (RMSE)

Le RMSE est la racine carrée de la variance des résidus. Il indique l'ajustement absolu du modèle aux données, à quel niveau les points de données observés sont proches des valeurs prédites du modèle. Le compare les valeurs prédites $h_\theta(x^i)$ et les valeurs de références y^i. Le RMSE est une formule populaire pour mesurer le taux d'erreur d'un modèle de régression. Il est donné par la formule

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(h_\theta(x^i) - y^i)^2}{n}}$$

3.6 Relative Squared Error (RSE)

Contrairement à la RMSE, l'erreur quadratique relative (RSE) peut être comparée entre les modèles dont les erreurs sont mesurées dans les différentes unités. L'erreur quadratique relative prend l'erreur quadratique totale et la normalise en divisant par l'erreur quadratique totale du prédicteur simple de la manière suivante :

$$RSE = \sqrt{\frac{\sum_{i=1}^{n}(h_\theta(x^i) - y^i)^2}{\sum_{i=1}^{n}(h_\theta(x^i) - Y)^2}}$$

et $Y = \frac{1}{n}\sum_{i=1}^{n} y^i$

DEUXIÈME PARTIE

Machine Learning & Sécurité

Chapitre 4

État de l'art de l'utilisation du Machine Learning à la Sécurité

Dans ce chapitre, il est question de faire l'état de l'art des différents domaines de sécurité où les techniques du Machine Learning ont fait leurs preuves. A travers cet état de l'art, nous allons explorer la question de la sécurité à travers la biométrie, la détection des botnets, la détection des tunnels DNS, la détection des **D**omain **G**enerating **A**lgorithm (DGA), la détection des URLs malicieuses, la détection des injections SQL, la corrélation des alertes, la détection des spams et la détection d'intrusion et ainsi que la détection des anomalies.

4.1 La biométrie

Face aux problèmes de sécurité causés par la fraude des documents, le vol d'identité, aux nouvelles menaces telles que le terrorisme [57] et la cybercriminalité, de nouvelles solutions technologiques sont mises en œuvres afin de garantir la sécurité des utilisateurs. Parmi ces technologies, il y'a la biométrie.

La biométrie est également appelée la mesure du corps humain. Cette mesure peut être physiologique (comme l'usage des empreintes digitales, la forme de la main, du doigt, le réseau veineux, l'iris et rétine, ou encore la forme du visage, pour les analyses morphologiques). Elle peut être aussi comportementale (liée à la reconnaissance vocale, la façon de frapper au clavier d'un ordinateur, la façon de bouger les objets, les gestuelles, etc).

Dans le domaine de la sécurité informatique, la biométrie permet de réaliser plusieurs fonctions essentielles liées à la question de la sécurité. Il s'agit de l'identification (déterminer l'identité d'une personne à travers son visage, l'enregistrement de sa voix, etc) et aussi de

l'authentification (c'est-à-dire vérifier et comparer les données des caractéristiques provenant d'une personne, au modèle de référence biométrique de cette dernière). La biométrie s'est distinguée comme étant la plus pertinente méthode pour identifier les personnes de manière fiable et rapide, en fonction de caractéristiques biologiques uniques. Elle constitue donc une véritable alternative aux mots de passe pour sécuriser les contrôles d'accès.

Dans le cadre de ce travail, nous explorerons deux branches de la biométrie (physiologique et comportementale) à savoir l'identification par empreinte digitale et l'authentification par keystroke. La raison de ce choix est dû à la richesse de la documentation et la clarté des approches ainsi que les différentes applications concrètes en lien avec la sécurité telles que les problèmes des mots de passe qui seront abordés dans ce rapport.

4.1.1 L'empreinte digitale

L'empreinte digitale est une branche de la biométrie qui permet de faire la reconnaissance ou l'identification d'une personne à travers ses doigts. Pour rappel, une empreinte digitale est constituée d'un ensemble de lignes localement parallèles formant un motif unique pour chaque individu. Elle comporte des parties à savoir : *les stries*[1] et *les sillons*[2]. Les stries ont en leur centre un ensemble de *pores* régulièrement espacés. Chaque empreinte possède un ensemble de points singuliers globaux (les centres) et locaux (les minuties).

Les méthodes classiques d'identification par empreinte digitale consistent à la classification basée sur des motifs de crêtes sur les doigts afin de classer les empreintes selon 5 points en plusieurs modèles à savoir : *arc*, *boucle*, *spirale*, *tendend arch* et *double boucle* [38]. Les objectifs de sécurité visés par l'identification par empreinte digitale sont liés aux problèmes de la sécurisation des mots de passe. L'identification via le mot de passe s'est avérée être un mécanisme de sécurité assez faible qui présente un risque élevé de fuite d'information (vol de mot de passe, vol des données, etc). Il existe plusieurs techniques afin de casser ou voler le mot de passe d'un utilisateur. Ces techniques sont par exemples, les attaques par force brute, les attaques par dictionnaire, *les key loggers* [3] et les moyens classiques tels que l'ingénierie sociale et l'espionnage.

La sécurisation des mots de passe est donc un enjeu considérable dans le domaine de la cybersécurité. Face aux problèmes liés à la gestion des mots de passe, l'identification par

[1] Les stries sont les lignes en contact avec une surface au toucher

[2] Les sillons sont les creux entre deux stries

[3] les key loggers sont des logiciels qui sont installés sur le poste de l'utilisateur, afin d'enregistrer les frappes de claviers effectuées par l'utilisateur

empreinte digitale permet de faire abstraction des mots de passe et tous les problèmes liées à leurs gestions et ainsi offrir une alternative d'identification unique pour chaque personne. Les solutions traditionnelles de problème d'identification des empreintes digitales sont de faire une classification basée sur les motifs généraux de plusieurs doigts, comme la présence ou l'absence de motifs circulaires pré-cités.

L'idée de l'utilisation du Machine Learning dans le cadre de l'identification par empreinte digitale est de construire un vecteur de caractéristiques basé sur les empreintes digitales pixelisées et utilisées afin d'entraîner un algorithme d'apprentissage à détecter les patterns cachés ainsi parvenir à matcher la bonne empreinte digitale au bon détenteur de l'empreinte.

Une quantité importante de recherches rapportées dans la littérature sur l'identification des empreintes digitales est consacrée aux techniques d'amélioration de l'image. Au sein des familles des algorithmes d'apprentissage du Machine Learning, force est de constater que le SVM et le réseau de neurones sont des algorithmes les plus utilisés dans l'identification par empreinte digitale [27] [65] [67]. La raison de l'utilisation du SVM peut s'expliquer par le fait qu'il fonctionne par la détection des patterns d'empreinte digitales et aussi sa forte résistance aux bruits. Comme nous l'avons mentionné précédemment, l'identification par empreinte vise à détecter le pattern d'une empreinte digitale parmi les autres patterns qui ne doivent pas nécessairement appartenir à la même classe. Le processus d'identification commence par une phase de conversion des données de scannes des empreintes digitales afin d'extraire les valeurs des pixels de niveau de gris allant de 0 à 255. Une phase d'harmonisation des images permet d'avoir la même quantité d'information sur chaque pixels. En d'autre termes, toutes les images peuvent être converties en gris.

Une phase de normalisation permet de régler, d'harmoniser les zones extrêmes (très blanches ou très sombres). La normalisation fonctionne par pixel et elle est utilisée pour standardiser les valeurs d'intensité dans une image en ajustant la plage de valeurs de niveau de gris. Chaque image est divisée en blocs et la variance d'échelle de gris est calculée pour chaque bloc de l'image et ainsi chaque image d'empreinte digitale est représentée par une matrice de **ligne** x **colonne**. Il est donc nécessaire d'avoir une matrice de même taille pour toutes les images d'empreintes digitales. D'autres méthodes existent pour extraire des caractéristiques telle que l'approche proposée par [65], qui par exemple utilise la méthode du *Gabor filter Bank* pour l'extraction des caractéristiques. Le filtre de Gabor est une structure mathématique qui permet de structurer une image afin de mettre en évidence les différentes formes, tailles et niveaux de douceur dans l'image.

L'identification par empreinte digitale est un problème non-linéaire, à cet effet, il faut donc choisir un noyau non-linaire afin de représenter les données. Le SVM calcule la distance entre les différents points représentant les différentes empreintes digitales pour déterminer les Hyperplans afin de séparer les différentes classes. [27] par exemple opte pour l'utilisation du SVM avec Radial Basis Function comme noyau. Pour rappel, le *Radial Basis Function* est un moyen d'approximer les fonctions multivariées par des combinaisons linéaires de termes basés sur une seule fonction univariée.

Dans la littérature, le réseau de neurones est largement utilisé dans le processus d'identification par empreinte digitale. [33] propose une implémentation en utilisant le réseau de neurones artificiels multi-couches pour fournir un algorithme de correspondance efficace pour l'authentification d'empreintes digitales. En utilisant la technique de rétro-propagation, l'algorithme fonctionne pour faire correspondre douze paramètres d'empreintes digitales et les relier à un nombre unique fourni pour chaque utilisateur autorisé. Les images de taille 188 $\times$ 240 pixels des empreintes digitales ont été utilisées sur un échantillon de 100 individus. Le réseau de neurones calcule la sortie de chaque couche, en extrayant l'erreur quadratique moyenne et en la propageant vers l'arrière si elle n'approche pas les cibles. Leur système a obtenu un taux de reconnaissance de 100%.

Une autre approche proposée par [9] pour la vérification des empreintes digitales. La vérification des empreintes digitales est basée sur les extrémités (minuties). Ainsi lorsqu'un nombre suffisant de minuties dans deux empreintes digitales s'accordent alors les deux empreintes digitales sont supposées identiques en calculant la distance euclidienne entre les caractéristiques. Le SVM utilisé par [9] a permis d'améliorer les taux d'erreur de 30%.

4.1.2 Le keystroke

Le keystroke est une branche comportementale de la biométrie. L'aspect comportemental vise à étudier la manière dont l'utilisateur interagit avec les touches afin de récolter les informations nécessaires telles que les différentes durées. Le keystroke vise à associer un aspect comportemental au processus d'authentification classique.

L'authentification par keystroke repose sur l'hypothèse selon laquelle les individus tapissent de différentes manières sur un clavier. L'authentification classique sur base de login et mot de passe présente des faiblesses dû à la compromission des mots de passe ou des clés (dans le cadre de la cryptographie). L'aspect comportemental permet d'assurer un deuxième niveau de sécurité. Un imposteur peut bien avoir le mot de passe d'un utilisateur mais ne peut pas s'authentifier étant donné qu'il ne possède pas les caractéristiques propres à l'utilisateur

légitime [23]. Le keystroke se réfère aux motifs habituels ou aux rythmes qu'un individu présente lors de la saisie sur un périphérique d'entrée. La manière dont un utilisateur manipule les touches peut être caractérisée à l'aide de la temporisation (comme le temps d'arrêt pour chaque touche, la vitesse entre les touches consécutives etc.).

Lors de la manipulation des touches, trois mesures importantes peuvent être prises en compte à savoir : le temps entre le moment où on appui sur une touche et le temps où cette touche est relâchée (Dwell Time), le temps qui sépare le relâchement d'une touche la pression sur une autre touches (Flying Time) illustré par la figure 4.1, ainsi que le temps séparant la pression sur deux touches consécutives appelé **Digraph**. Rappelons que lorsqu'une touche est pressée, elle crée une interruption matérielle pour le processeur et génère un **timestamp** mesuré avec une précision en microseconde. En plus des caractéristiques standards vues précédemment, on peut également utiliser des outils permettant de discerner les utilisateurs. [47] acquiert la mesure des durées de touche et la mesure de force exercée sur les touches via un capteur de force.

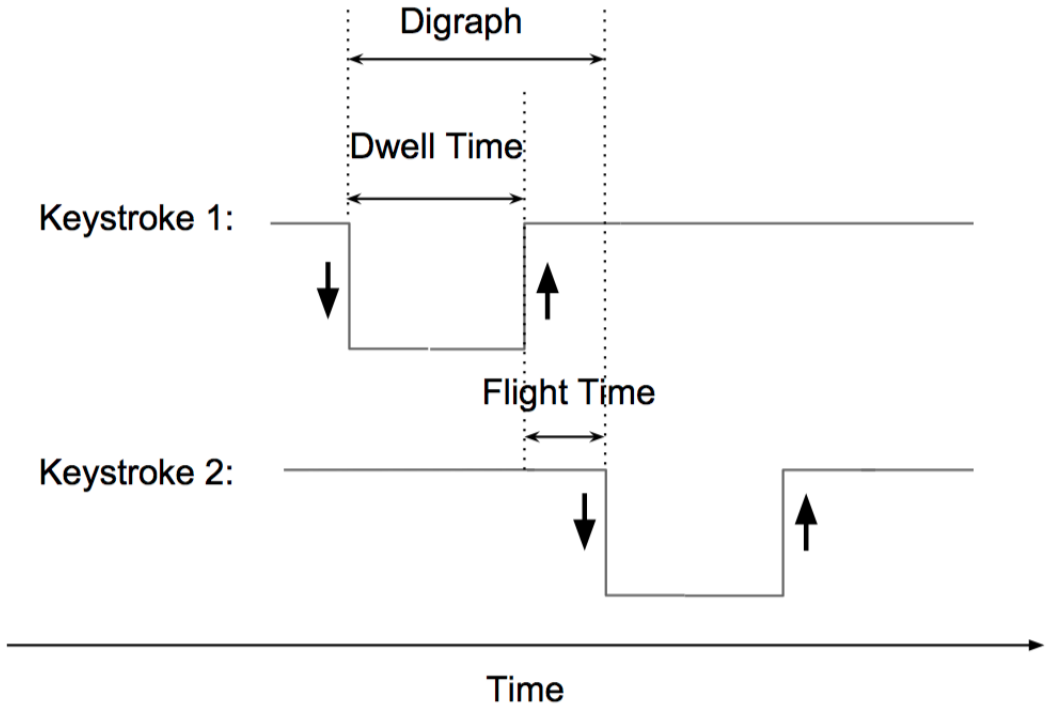

FIGURE 4.1 – La liste des caractéristiques proposée par [23].

On remarque qu'il n'y a pas assez de diversité de données, car plusieurs auteurs utilisent

les données du *CMU keystroke Dynamics Benchmark*. Cette source de données des caractéristiques précédemment citées relatives aux 51 utilisateurs tapant au clavier 400 fois le même mot de passe **".tie5Roanl"** [23].

Dans les écrits, l'authentification par keystroke est un problème approché dans le cadre de l'utilisation du Machine Learning par le mode d'apprentissage supervisé. Il s'agit de valider ou rejeter un utilisateur sur base des caractéristiques des frappes effectuées. Plusieurs algorithmes de type classification peuvent être employés à cet effet. Dans les travaux plusieurs approches peuvent être observées, l'approche [104] analyse les performances du KNN, C4.5 et le Naïve Bayes. [47] choisi le SVM comme l'algorithme d'apprentissage pour créer le modèle. La fonction de noyau du SVM est le type polynomial de degré 5 ainsi que le taux de pénalité fixé à 100. A travers ce choix, [47] cherche à construire un hyperplan non-linéaire de degré 5 afin de séparer les deux classes. Le paramètre C mesure le compromis entre l'erreur de formation et la maximisation de la marge, la valeur élevée du paramètre C permet de réduire la marge de l'hyperplan dans chaque classe.
Différentes mesures de distance, telle que la distance euclidienne a été utilisée par [13] pour mesurer la similarité des caractéristiques. [62] estiment que l'utilisation du KNN afin de trouver des similarités par rapport à un point pose des problèmes de performances car l'algorithme du KNN parcourt l'ensemble des points et calcule la distance par rapport à ce point afin de sélectionner les K-points les plus proches de ce point. Le parcours de tous les points est coûteux en terme de temps lorsqu'on est en présence d'une grande quantité de données des utilisateurs.

4.2 La détection des botnets

L'un des problèmes en matière de sécurité des réseaux informatiques sont les botnets. Les botnets peuvent êtres définis comme un groupe coordonné d'instances de logiciels malveillants contrôlés via des canaux de communication (C&C) par un botmaster [64]. Le canal de commande et communication permet ainsi la coordination des botnets. Ils forment un réseau massif d'ordinateurs compromis utilisés pour attaquer d'autres systèmes informatiques et constituent également la plate-forme clée pour de nombreuses attaques sur internet telles que les spams, le déni de service (DoS), le vol d'identité, le phishing, etc.

Les botnets utilisent les protocoles HTTP (Hypertext Transfer Protocol), IRC (Internet Relay Chat, le protocole IRC est un protocole de communication textuelle sur Internet), P2P (Pair to Pair) ainsi que le FTP (File Transfer Protocol) pour télécharger le code malveillant à partir d'un emplacement réseau externe et s'installe sur la machine vulnérable [64].

Une fois chargée sur un ordinateur client, le bot compromet la machine vulnérable en utilisant le canal C&C, la met sous la commande de l'attaquant [64]. Ainsi le client infecté devient un zombie ou bot à son tour. Les attaques des botnets sont possibles en utilisant des milliers et même des millions d'ordinateurs compromis ainsi l'impact de l'attaque est énorme [107]. L'utilisation d'un botmaster de canal C&C déployé peut contrôler à distance le comportement du bot malware, rendant le fonctionnement du bot plus flexible et adaptable aux besoins du botmaster. Le cycle de vie d'un bot est généralement constitué de trois phases à savoir : la phase d'infection, de communication ainsi que la phase d'attaque.

La phase d'infection est la première phase du cycle de vie d'un botnet selon laquelle les ordinateurs vulnérables sont compromis par le bot malware, devenant ainsi des zombies à leur tour. L'infection peut être réalisée de différentes manières, telles que par le téléchargement indésirable de logiciels malveillants sur des sites malveillants, le téléchargement de fichiers infectés joints à des messages électroniques, la propagation de logiciels malveillants à partir des supports tels que les USB et disques. Elle couvre la communication consacrée à la réception des instructions et des mises à jour du botmaster.

La deuxième phase tient compte des modes d'opérations des botnets en impliquant communication entre des ordinateurs infectés et des serveurs C&C. La communication entre zombie et le serveur est réalisée en utilisant le canal C&C.

Pendant la troisième phase, l'ordinateur zombie effectue des actions à son tour telles que lancer des attaques DoS, lancer des campagnes de courrier indésirable, distribuer les identités volées, déployer la fraude au clic ainsi que faire du fast-flux. Le fast-flux est une technique DNS utilisée par les botnets pour cacher les sites de diffusion de logiciels malveillants et les sites de phishing derrière un réseau en constante évolution d'hôtes compromis agissant en tant que mandataires.

La détection des botnets peut se faire par les signatures ou via la méthode de la détection d'anomalies [107]. De nombreuses techniques de détection de botnets sont basées sur des signatures d'attaques. Le système basé sur des signatures détecte des botnets connus (dont la signature est connue), ils souffrent d'un inconvénient majeur car ils sont incapables de détecter les variantes des signatures des attaques et présentent également un taux de faux positifs élevé [61]. La détection des botnets peut se faire sur les postes clients ou sur le réseau. La détection des botnets sur les postes clients vise à détecter la présence de logiciels malveillants en examinant différentes analyses au niveau du client, telles que les journaux

des applications, les processus actifs, les journaux de clés, l'utilisation des ressources.

La seconde approche basée sur la détection réseau permet la détection de réseaux de zombies en analysant le trafic réseau. Cette technique identifie les botnets en reconnaissant le trafic de réseau produit par eux dans les trois phases du cycle de vie des botnets pré-citées. L'application des techniques du Machine Learning face à la détection de botnets présume que les botnets produisent des patterns (modèles) qui peuvent être détectés en utilisant des algorithmes du Machine Learning tels que l'arbre de décision, le SVM, le Naïve Bayes. ainsi que le KNN.

Plusieurs auteurs ont exploré la détection des botnets véhiculés à travers le protocole P2P. [74] propose une analyse du comportement du réseau pour la détection des botnets utilisant le protocole P2P. Un certain nombre de caractéristiques au niveau du paquet comme l'adresse IP source, le numéro de port source, l'adresse IP de destination, le port de destination, le protocole utilisé ainsi la durée de la connection sont identifiées et ensuite utilisées pour distinguer le trafic botnet du trafic normal. Des algorithmes d'apprentissage vue dans la première partie tels que l'arbre de décision, le classificateur KNN ainsi que le SVM ont été utilisés pour détecter les botnets comme dans la plupart des travaux.

Sur un jeu de données des paquets TCP issu du trafic des botnets, [74] extrait les caractéristiques suivantes : le port de source, le port de destination, l'adresse source, l'adresse cible, le nombre de bytes ainsi que la durée de la connexion. Les résultats obtenus via l'exactitude des classificateurs tel que la précision du classificateur KNN varie entre 33,3 % et 97,10% pour différentes classes de trafic botnet tandis que l'arbre de décision et SVM ont atteint des résultats près de 100% corrects. Précisons que ces résultats sont obtenus en utilisant seulement une fraction de l'ensemble de données pour une étude préliminaire. [61] propose un système de détection de botnet en 3 phases : la collection de données, l'extraction des caractéristiques ainsi que la phase de test. Les données sont capturées via Wireshark (analyseur de protocole réseau). A travers les données capturées, [61] extrait les informations relatives aux adresses IP (sources et cibles) ainsi que les ports comme l'a fait [74]. En plus les informations relatives aux paquets sont également pris en compte telles que les accusées de réceptions entre deux hôtes. [61] opte pour une comparaison entre KNN, le réseau de neurones multi-couches, ainsi que Random Forest (arbre aléatoire).

Le protocole HTTP est fréquemment utilisé par les botnets en tant que protocole de commande et de communication. En effet l'une des raisons est le fait que les trafics HTTP sont les plus souvent autorisés afin d'accéder aux sites internet et ne présentent pas souvent de

danger. Autrement dit, les botnets basés sur HTTP sont difficiles à détecter car leur trafic C&C peut se cacher derrière un trafic Web normal pour échapper aux mécanismes de détection [64]. Les botnets utilisent ainsi le protocole HTTP afin de dissimuler leurs activités malicieuses. [8] propose un système d'analyse des botnets implémentés en utilisant le classifier l'algorithme C5.0 basé sur les arbres de décision, ainsi le K-Mean en se basant sur les entrées générées par les machines qui peuvent contenir des malwares.

Le Machine Learning est une solution pour détecter différents types de botnets car il fournit une solution fiable pour détecter la présence des botnets en se basant sur la reconnaissance des formes [107]. [64] propose une comparaison sur les performances de l'arbre de décision (C4.5), du réseau bayésien et le SVM en utilisant des mesures de performance telles que la précision, la sensibilité, la valeur des vrais positifs dans le cadre de la détection des botnets. Les données ont été collectées aux quelles des caractéristiques ont été extraites des en-têtes de paquets HTTP. La validation 10-fold cross a été utilisée afin de tester les modèles. Les résultats montrent un taux élevé des VP et un très taux faible de FP pour les meilleurs modèles obtenus. Un taux positif réel élevé signifie que les classificateurs d'apprentissage automatique ont bien fonctionné dans la prédiction des flux des bots réels.

4.3 La détection des tunnels DNS

Le tunnel DNS est une méthode de cyberattaque qui permet encoder les données d'autres programmes dans les requêtes et les réponses DNS. Il peut être vu comme étant un moyen de détourner le protocole DNS pour l'utiliser comme un protocole de communication secret ou un moyen d'exfiltration de données [1]. Le tunnel DNS permet à des cybercriminels d'insérer des logiciels malveillants (section 4.2) ou de transmettre des informations volées dans les requêtes DNS, afin de créer ainsi un canal de communication dissimulé qui contourne la plupart des pare-feux.
On distingue principalement quatre types de tunnels DNS : le tunnel DNS à travers HTTP, HTTPS, FTP ainsi que POP3 [1]. Les techniques des détections des tunnels DNS via les algorithmes du Machine Learning peuvent être classées en deux types d'analyses à savoir l'analyse du trafic ainsi que l'analyse de la charge utile (payload) [53]. L'analyse de trafic analyse le trafic de manière globale afin d'extraire certaines caractéristiques significatives pouvant telles que le volume de trafic DNS, le nombre de noms d'hôtes par domaine, l'emplacement et l'historique du domaine tandis que l'analyse de la charge utile analyse la charge utile d'une seule demande afin d'identifier des caractéristiques telles que la longueur du domaine, le nombre d'octets et le contenu [53]. Parmi les caractéristiques, [53] extrait les caractéristiques telles que la longueur de paquet d'adresse IP, la longueur de réponse de paquet IP

et la longueur de nom de requête DNS codée. La détection des tunnels DNS est abordée dans la littérature comme un problème de classification. En effet, il s'agit d'identifier un ensemble de données qui contient des connexions DNS avec leur étiquette de classe correspondante (positive ou négative). Plusieurs approches de détections de tunnel ont étés examinées via les algorithmes du Machine Learning.

Pour détecter les tunnels DNS, les auteurs [25] ont présenté une approche en utilisant plusieurs algorithmes de type classification tels que le Random Forest et l'arbre de décision sur une base de données composée de plus de 20 000 flux capturés sur un réseau réel. Le modèle Random Forest est utilisé pour déterminer la classe de chaque connexion (HTTP, HTTPS, POP3, SSH, SMTP, etc).

[1] propose quant à lui une approche détection de tunnel DNS en utilisant le SVM multi-classification afin de classifier les différents type de tunnels DNS sur un jeu de données contenant 106 enregistrements de chaque tunnel DNS ainsi que 106 enregistrements des traffic normaux. Diverses caractéristiques ont été extraites telles que la longueur de la requête DNS, la longueur de l'émetteur IP, la longueur de la réponse IP, la longueur du nom de la requête DNS codée, l'entropie de la couche application, l'entropie des paquets IP et l'entropie du nom de la requête.

4.4 La détection de DGA

Les cybercriminels détournent le protocole DNS et l'utilisent à leurs propres fins. Les algorithmes de génération est une série d'algorithme qui génère automatiquement les noms de domaines aléatoire, puis générer une liste de candidats C&C domaines [96]. L'algorithme produit des noms de domaine à apparence aléatoire pouvant être utilisés comme points de rendez-vous avec leurs serveurs de commande et de contrôle. L'idée est que deux machines utilisant le même algorithme vont contacter le même domaine à un moment donné, afin qu'elles puissent échanger des informations ou récupérer des instructions. Selon [34], certains DGA n'utilisent que la date actuelle comme graine (seed) permettant la génération automatique des noms de domaines.

Les algorithmes de génération de domaine (DGA) sont des techniques répandues pour établir la communication de l'intérieur du réseau compromis vers l'un des domaines contrôlés par les acteurs de la menace. Plusieurs logiciels malveillants utilisent des algorithmes de génération de domaine (DGA) pour établir des connexions de commande et de contrôle pour communiquer avec un botmaster [12] [26]. Les logiciels malveillants qui utilisent des DGAs

sont entre autres *Cryptolocker, Dircrypt*, *Bobax*, *Kraken*, *Corebot*, etc. Ils utilisent des algorithmes de génération de domaine afin de générer un grand nombre de domaines [26] par jour afin d'éviter d'être détectés par des systèmes de détection de malwares basés sur la détection statique telle que la détection via la liste noires. Si un ou plusieurs noms de domaine C&C sont identifiés et supprimés, les bots vont déplacer le nom de domaine C&C via des requêtes DNS vers l'ensemble suivant de domaines générés automatiquement [19].

Les domaines provenant de la liste Alexa [3] sont généralement utilisés comme exemples de domaines non malveillants dans les différentes approches. Les caractéristiques utilisées pour la classification des domaines crées par des algorithmes de génération de domaine varient de base, comme la longueur de la chaîne de caractères, le nombre de voyelles et de consonnes, à plus complexe, comme l'entropie et la probabilité conditionnelle de n-gramme [34]. En utilisant le caractère aléatoire de la génération des DGA, [96] propose une approche de détection des DGA en utilisant le Random Forest (forêt aléatoire), le SVM et le Naïve Bayes. ils utilisent 3,4,5-gramme afin de générer les noms d'extraire les caractéristiques a permis d'améliorer la précision de la classification. L'experience a montré que le Random Forest a obtenu de très bons résultats dans la classification des DGA.

Une autre approche proposée par [55] *Peiades* suppose que la réponse aux requêtes DGA serait principalement un domaine non-existant (NX-Domain) et que les machines infectées dans le même réseau avec les mêmes DGA recevraient une réponse d'erreur NX-Domain similaire. Les auteurs [55] ont combiné des techniques de classification et de clustering pour regrouper des domaines similaires par le modèle de chaîne de nom de domaine et ensuite définir le DGA auquel ils appartiennent. Leur système présente des taux de vrais positifs de 95 à 99% et des taux de faux positifs de 0,1 à 0,7%. [45] [44] quant à eux proposent une vision différente dans la détection des domaines générés automatiquement. En effet, leur approche exploite les réseaux de neurones à mémoire à long-court terme (LSTM) pour la prédiction des DGA sans avoir besoin d'informations contextuelles ou des caractéristiques pré-citées et en se basant uniquement sur des noms de domaine bruts. La technique LSTM mise en place peut fournir un taux de détection de 90% avec un taux de FP de l'ordre de 1 sur 10000.

4.5 La détection des URLs malicieuses

On appelle **U**niform **R**esource **L**ocator (URL) l'adresse globale des documents et autres types de ressources sur le web. Une URL se compose de l'identifiant de protocole qui indique quel protocole utilisé et le nom de ressource qui spécifie l'adresse IP ou le nom de domaine où se trouve la ressource [20].

Les URLs malveillantes hébergent du contenu non sollicités qui incitent les utilisateurs à devenir victimes d'escroqueries. Afin de protéger les utilisateurs, il est indispensable de détecter ces menaces. Ces attaques sont généralement menées en exploitant les vulnérabilités dans les plugins ou en insérant du code malveillant. Les menaces incluent des sites Web frauduleux qui incitent les utilisateurs à révéler des informations sensibles qui finissent par entraîner le vol d'argent ou d'identité, ou même l'installation de logiciels malveillants sur la machine de l'utilisateur. Afin de lutter contre ce type d'attaque, il existe deux approches de détections des URLs malicieuses à savoir la détection via la liste noire ainsi que la détection de URLs malsaines via les techniques du Machine Learning.

La liste noire constitue une base de données d'URLs qui ont été confirmées comme malveillantes au paravant [60]. Cette base de données est continuellement mise à jour au fur et à mesure qu'une nouvelle URL malicieuse est connue. Il est impossible de maintenir une liste noire d'URLs malveillantes, d'autant plus que de nouvelles URLs sont générées quotidiennement. Les hackers utilisent des techniques pour échapper aux listes noires et tromper les utilisateurs en modifiant l'URL pour paraître légitime. Force est de constater que la détection par la liste noire ne peut pas détecter de manière exhaustive les URLs malicieuses compte tenu du fait que les ULRs malsaines doivent être tenues à jours lorsqu'une nouvelle URL malsaine a été découverte.

Afin d'améliorer la détection des URLs malveillants, les techniques du Machine Learning ont été utilisées à cet effet. Les approches d'apprentissage automatique, utilisent un ensemble d'URLs comme données d'apprentissage qui contiennent des URLs saines et malicieuses. En fonction des caractéristiques, elles apprennent à faire des prédictions pour classer des URLs, ainsi pour généraliser à de nouvelles URLs contrairement aux méthodes de liste noire.

Les données d'apprentissage peuvent être collectées sur Phishtank [60] contenant une liste d'URLs jugées malicieuses. Cette liste est tenue continuellement à jour lorsqu'une nouvelle URL malsaine a été découverte. Une fois les données d'apprentissage collectées, il faut extraire les caractéristiques informatives afin qu'elles décrivent suffisamment les URLs.

Les caractéristiques lexicales se rapportent aux propriétés textuelles des URLs telles que la présence d'une adresse IP dans l'URL, la taille de l'URL, le nombre de slashes et de points dans l'URL, etc. D'autre types de caractéristiques relatives à l'hôte (les informations WHOIS), telles que les informations sur l'hébergeur, ainsi que les informations sur les dates (d'enregistrement, de mise à jour et d'expiration du domaine), la localisation [20], etc. Ces carac-

téristiques doivent être converties dans un format spécifique par exemple dans un vecteur numérique afin d'être utilisées par un algorithme d'apprentissage du Machine Learning qui puisse former un modèle d'apprentissage. La détection d'URLs malicieuses est un problème de classification binaire car il s'agit de classer les URLs en URLs saines ou malicieuses sur base des caractéristiques des URLs extraites. Plusieurs algorithmes de classification ont été utilisés sur les données d'entraînement afin de construire les modèles de détection. Parmi ces algorithmes on distingue entre autres : le SVM, la RL, le Naïve Bayes ainsi que l'arbre de décision [106]. [72] propose un mécanisme en utilisant les algorithmes d'apprentissage automatique de classification tel que le SVM pour détecter les URLs courtes malveillantes avec des caractéristiques lexicales [5], le contexte de tweet et les caractéristiques sociales d'un Twitter social en ligne populaire. [106] propose une approche basée de détection des URLs malicieuses en utilisant l'algorithme C4,5 sur 30 caractéristiques incluant les caractéristiques statiques ainsi que les informations relatives au WHOIIS.

[28] aborde la détection d'URLs malveillantes comme étant un problème de classification binaire en étudiant les performances de plusieurs classificateurs tels que le Naïve Bayes, SVM, l'arbre de décision ainsi que le KNN en utilisant une grande quantité d'URLs ainsi qu'une grande quantité de liste noire comme caractéristiques.

4.6 La détection des injections SQL via Machine Learning

L'injection **S**tructured **Q**uery **L**anguage (SQL) est une technique d'injection qui permet d'attaquer des sites internets en insérant des instructions SQL à travers des paramètres d'entrée afin de modifier la requête. Cette requête peut être malveillante et poser des problèmes aux applications et permettre à l'attaquant de s'introduire dans le système. A travers des failles découvertes dans le système Il faut détecter cette injection avant qu'elle ne soit exécutée. En exploitant les attaques par injection SQL, il est possible de voler des informations importantes telles que les mots de passe bancaires, les identifiants des applications Web et même de supprimer des données de la base de données [7].

Les types de requêtes utilisés pour les injections SQL sont les suivants : les tautologies sont utilisées pour contourner l'authentification, requêtes logiquement incorrectes, et l'union requêtes utilisée pour acquérir des informations utiles de la base de données, des procédures stockées sont utilisées pour stocker la requête malveillante dans la base de données [7].

Plusieurs techniques du Machine Learning ont été employées dans la détection des injections SQL. [7] propose une approche en utilisant la base de données Postgres SQL pour générer

l'arborescence de requête des instructions SQL. Les caractéristiques sont extraites des entités de l'arbre de requête à travers le score de Fisher. La différence entre l'arborescence de requête normale et l'arborescence de requête malveillante peut être mesurée en sélectionnant des entités spécifiques dans l'arborescence de la requête. L'algorithme de classification SVM a été employé afin d'atteindre une précision de 94% en utilisant 10-fold cross-validation.

[24] propose un système de prévention des injections SQL basé sur le Machine Learning. Le système proposé capture les paramètres des requêtes HTTP comme caractéristiques numériques. Les caractéristiques numériques incluent la longueur des paramètres et le nombre de mots-clés des paramètres. Les mots-clés sont des instructions SQL telles que **SELECT, FROM,*, etc**. A travers ces caractéristiques, le système classifie les paramètres par un classificateur Bayésien pour juger si les paramètres sont des modèles d'injection. Le système a été évalué avec un outil d'attaque par injection SQL populaire nommé SQLMap. Les résultats de l'évaluation montrent que le système proposé est capable d'empêcher une attaque par injection SQL et un taux de détection positif élevé.

[76] propose une approche de détection des injections SQL en utilisant le SVM. Chaque mot extrait de la chaine de requête constitue une caractéristique. Le système de détection proposé a atteint une précision de l'ordre de 94,47%.

[46] propose une approche de détection des injections SQL en utilisant l'arbre de décision (J48). Leur approche utilise le trafic capturé entrant dans l'application Web en combinaison avec le trafic capturé entre l'application Web et le serveur de base de données associé pour classer le trafic entrant comme étant normal ou malveillant. Le résultat de différentes experiences ont permis d'avoir des scores de détections des injections SQL supérieures à 80% sur plusieurs sources de données différentes.

4.7 La détection des spams

Le spam est une technique de prospection qui consiste à diffuser massivement par courrier électronique des informations, souvent de nature publicitaire, non sollicitées par les internautes destinataires. Ces spams sont envoyés à des millions d'adresses emails sans qu'ils aient été sollicités. Le spam peut par exemple contenir des scripts potentiellement dangereux, des virus ou équivalent qui s'activent sur un système. La lutte contre les spams est une activité où le Machine Learning a contribué à detecter les courriers indésirables de manière remarquable. Différentes méthodes ont été développées pour filtrer le spam, notamment la liste noire, l'appariement des mots clés . Plusieurs techniques largement utilisées com-

prennent le classificateur d'arbre de décision C4.5, perceptron multi-couches et classificateur Naïve Bayes, qui sont régulièrement les plus utilisés cependant l'algorithme de classification Naïve Bayes reste très largement utilisé.

Au niveau de l'extraction des caractéristiques, on distingue deux sortes d'approches : l'extraction des caractéristiques basée sur le sac de mots [95] prédéfini et l'extraction des caractéristiques basée sur la fréquence d'apparition des mots. Au sein des approches utilisant le sac de mots, les caractéristiques extraites à base de l'algorithme sont n-gramme extraites en sélectionnant n mots contigus d'une séquence données. L'extraction des caractéristiques basée sur la fréquence d'apparition des mots sera abordée de façon approfondie dans la partie contribution de ce mémoire.

Les textes d'emails sont convertis en vecteurs de caractéristiques. Par exemple, il est très courant de prendre le vecteur des nombres d'occurrences de certains mots dans un message en tant que vecteur de caractéristiques utilisant le sac de mots. [52] propose une technique pour classer les emails de phishing en fonction des propriétés structurelles des courriels de phishing. Ils ont utilisé un total de 25 caractéristiques. Ils ont testé 200 emails (100 emails phishing et 100 emails légitimes). Après avoir choisi un ensemble de fonctionnalités, ils ont utilisé le gain d'information pour classer ces caractéristiques en fonction de leur pertinence. Ils ont appliqué une SVM pour classer les emails de phishing en fonction des fonctionnalités sélectionnées. Leurs résultats indiquent un taux de détection de 95% des courriels de phishing avec un faible taux de faux positifs.

[4] propose un modèle comparatif en utilisant 3 algorithmes de classifications le Naïve Bayes, l'arbre de décision (C4.5) et le réseau de neurones. Les données d'apprentissage ont été testées en termes d'informations d'en-tête de message, de liste noire et blanche et de revue de mot clé. L'exactitude du modèle conçu avec le Naïve Bayes a atteint un taux de détection de 98,5% ainsi que l'arbre de décision avec une exactitude de 96,6% et le réseau de neuronal avec 99,9%.

4.8 La détection d'intrusion

L'intrusion est définie comme toutes actions ou tentatives visant à la compromission, à l'intégrité, à la confidentialité et à la disponibilité d'une ressource. Elle est une tentative délibérée et non autorisée d'accéder, de manipuler des informations ou de rendre un système inutilisable via les attaques [42]. Il existe plusieurs sortes d'attaques reparties en 4 groupes à savoir : **les attaques DOS, R2L, U2R et le sniffing**.

Les attaques DOS (Deny Of Service) ou déni de service est un type d'attaque où un hacker effectue des manipulations sur des ressources informatiques pour en saturer la mémoire via des requêtes. La machine sera alors très occupée pour ne pas traiter des demandes légitimes. Les utilisateurs légitimes sont interdits d'accéder à la machine.

Les attaques de type U2R (User To Root) sont des types d'attaques où un simple utilisateur ayant l'accès à un compte d'utilisateur normal sur le système est capable d'exploiter la vulnérabilité pour accéder au système.

Les attaques de type R2L (Remote To Local user) sont des types d'attaques à distance dans lequel un attaquant envoie des paquets à une machine sur un réseau dont il n'a pas de compte. Il exploite une certaine vulnérabilité pour obtenir un accès local en tant qu'utilisateur de cette machine.

Le sniffing est une technique d'attaque ou le hacker scanne le réseau afin de détecter les vulnérabilités les plus connues sur les machines et les exploiter. Cette récolte des vulnérabilités peut se faire via l'ingénierie sociale (type manipulation psychologique à des fins d'escroquerie) ou des outils de scan tel que *Nmap*.

Les systèmes de détection d'intrusion classiques nécessitent l'apport humain pour différencier le trafic réseau intrusif et non intrusif. Des interventions humaines sont nécessaires pour créer, tester et déployer les signatures sur les ensembles de bases de données. Ainsi, cela peut prendre des heures ou des jours pour détecter et générer une nouvelle signature pour une attaque, ce qui peut impliquer un temps illimité pour faire face à des attaques rapides.

Afin de lutter contre les intrusions plusieurs méthodes de détection ont permis de détecter les intrusions. Les **I**ntrusion **D**etection **S**ystem (IDS) sont des logiciels ou des matériels physiques qui ont pour but de protéger les réseaux contre les attaques. Les IDS aident les réseaux à résister contre les attaques externes. Ils sont capables de donner des commandes de prévention aux pare-feux et d'apporter des modifications au contrôle d'accès aux routeurs. Ils peuvent prendre des décisions de contrôle d'accès en fonction du contenu de l'application plutôt que l'adresse IP ou les ports comme les pare-feux traditionnels [10]. Certains IDS surveillent un seul ordinateur (HIDS) en particulier en regardant les logs ainsi que toutes informations se trouvant sur l'ordinateur afin de déduire s'il y a eu attaque, tandis que d'autres sont capables de surveiller un réseau (NIDS) particulier ou de nombreux réseaux formant un réseau étendu et interceptent les paquets circulant dans un réseau.

Il existe également un autre type de détection d'intrusion appelé Hybrid IDS. Dans ce type d'IDS, les sources d'informations viennent à la fois des hôtes (HIDS) ainsi que du réseau (NIDS). Dans les trois cas, ils recherchent des signatures des attaques (modèles spécifiques) qui indiquent généralement une intention malveillante ou suspecte. [42] définie comme signature est un ensemble de règles qu'un IDS utilise pour détecter une activité intrusive typique, telle que les attaques DoS.

Les systèmes de détection d'intrusion détectent les intrusions sur base d'anomalie ou sur base des signatures. La détection d'intrusion basée sur les signatures (ou encore scénario) fonctionne en comparant les signatures observées aux signatures connues dans la base de données de signatures. Cette base de données est une liste de signatures des attaques connues antérieurement [42]. Elle présume l'existence d'une base de connaissances sur les attaques existantes et dès qu'une activité correspond à l'un des éléments de cette base de connaissances, alors une alerte sera générée. La détection par signature est très faible aux attaques *0days* [50] car ce sont les attaques qui sont exploitables dès le premier jours qu'elles apparaissent sur les sites de partages ainsi que dans les articles. Cependant elle est très efficace contre les attaques connues, mais elle ne peut pas détecter de nouvelles attaques tant qu'elles ne sont pas mises à jour avec de nouvelles signatures [59] [42]. La mise à jour des signatures est coûteuse et fastidieuse [10] car elle exige l'intervention d'un analyste de sécurité.

La détection d'intrusion sur base d'anomalie (comportementale) a pour but de selon [57] de prédire un comportement en utilisant une base de données de comportements normaux pour ainsi reconstituer le profil. Ainsi lorsque le comportement d'un profil diffère du comportement normal attendu alors ce comportement sera donc suspect. Un tel système de détection d'intrusion est donc défini comme une identification d'une déviation du comportement de l'utilisateur par rapport à son comportement normal. Grâce à l'approche comportementale des attaques inconnues peuvent être décelées. Selon [50], l'approche comportementale nécessite un temps d'apprentissage conséquent pour réaliser le profil des utilisateurs par contre, elle permet de détecter les attaques nouvelles [59].

Les algorithmes du Machine Learning ont contribué dans les travaux à detecter les intrusions. La détection d'intrusion via l'approche supervisée exige que les données soient classifiées normales ou anormales par un expert. Les IDS basés sur les signatures reposent sur la création des modèles capables de classifier les nouvelles signatures. Ainsi les modèles sont testés et déployés afin de classer les attaques tant disque les IDS basés sur des anomalies en utilisant des techniques les algorithmes du Machine Learning ont la capacité de mettre en

œuvre un système capable d'apprendre à partir de données et de regrouper les paquets selon leur similitude. L'utilisation du Machine Learning a pour but d'apporter une aide dans la prise de décision aux analystes dans la généralisation automatique des règles de détection. Ces techniques sont basées sur l'analyse statistique des données. La capacité de ces algorithmes qui traitent ces ensembles de données qui peuvent utiliser des modèles trouvés dans des données antérieures afin de prendre des décisions pour les nouveaux modèles de données en évolution dans le trafic réseau.

Les systèmes de détection d'intrusions basés sur les algorithmes du Machine Learning sont des systèmes qui apprennent à partir des comportements normaux et ainsi construire des modèles pouvant aider à classifier (catégoriser) un nouvel comportement.

Force est de constater qu'au niveau des données, la grande majorité des travaux effectués [70] [14] [59] [58] [86] ont utilisé le dataset de KDD CUP99. La raison principale de cette popularité est dû au fait que le dataset CUP 99 est riche en caractéristiques (42) et contient une diversité d'attaques reparties en 4 groupes pré-cités au début du chapitre. Ce dataset est constamment alimenté (de 400.000 instances en 1999 à 1.500.000 instances en 2000) [59]. Selon [14], les techniques de l'apprentissage automatique ont conduit à des méthodes améliorées d'analyse des problèmes de sécurité du réseau de plusieurs façons via la détection d'anomalie sur le système de réseau en utilisant des données statistiques pour calculer les problèmes de réseau. [14] proposent une étude comparative sur les algorithmes tels que SVM, l'arbre de décision (C4.5), le Naïve Bayes en utilisant le dataset de KDD CUP99.

[101] propose un NIDS basé sur le scan des ports en utilisant le SVM en collectant les paquets réseau . Les paquets sont capturés à l'aide des outils WINPCAP et JPCAP. La sélection et l'extraction des caractéristiques ont été réalisées via les méthodes du Rough Set et le PCA afin de récolter les informations des paquets réseau comprenant l'en-tête IP, l'en-tête TCP, l'en-tête UDP et l'en-tête ICMP de chaque paquet. Ainsi 29 caractéristiques telles que les numéros de port TCP et UDP source et destination ont été extraits. Les Rough Set sont des techniques d'extraction des caractéristiques capable de gérer les données hétérogènes. Cette approche commence par un ensemble vide de caractéristiques ce qui entraîne un accroissement maximum de la dépendance. La combinaison Rough Set et le SVM a permis d'obtenir la valeur d'exactitude de 93,33%.

[15] utilise l'algorithme génétique ainsi que l'arbre de décision (ID3) afin de générer les règles permettant de classer les connections réseaux basés sur leur signature. Les caractéristiques utilisées sont les adresses IP (source et destination), les ports (source et destination) ainsi que

le protocole de communication. [86] propose une méthodologie de détection d'intrusion par détection d'anomalie en utilisant les données du KDD-99 en testant les différents types des réductions des caractéristiques telles que la sélection et l'extraction des caractéristiques. L'extraction des caractéristiques est effectuée en utilisant l'analyse des composants principaux du noyau (KPCA une extension du PCA) en combinant le SVM a permis d'obtenir un niveau d'exactitude de 90,91 % ainsi qu'une précision de 84 %.

La détection d'intrusion sur base d'anomalie via l'approche non-supervisée permet de regrouper les paquets selon leur ressemblances. Cette vision est robuste car elle ne dépend pas de l'évolution des caractéristiques du traffic. Le K-Mean est l'algorithme de clustering le plus fréquemment utilisé dans la détection d'intrusion. L'un des avantages du K-Mean réside dans sa capacité à gérer les grandes quantités de données. Le K-Mean est utilisé pour grouper les données afin de trouver les structures ou les modèles significatifs dans une collection de données non-étiquetées, de sorte que les instances dans le même cluster sont similaires, tandis que les instances de différents clusters sont différentes les unes des autres. L'avantage de K-Mean est sa flexibilité dans le traitement de grands ensembles de données. ll crée K des groupes de points de données similaires. Les instances de données qui n'appartiennent pas à ces groupes peuvent être marquées comme des anomalies.

Les auteurs [66] ont proposé une méthode de détection d'anomalie utilisant K-Mean combinée avec C4.5 pour classer les activités anormales et normales. Dans cette approche, le clustering K-Mean est utilisé initialement pour partitionner le jeu de données d'apprentissage en K clusters en utilisant la distance euclidienne. Ensuite, l'arbre de décision est construit pour chaque cluster en utilisant l'algorithme du C4.5 et les règles créées par l'arbre de décision sont utilisées pour détecter les événements d'intrusion. La phase de test est mise en œuvre en deux étapes. La distance euclidienne est calculée pour chaque instance de test, avant de trouver le cluster le plus proche. Par conséquent, l'arbre de décision correspondant au cluster le plus proche est sélectionné pour détecter la classe de l'instance. Dans ce travail, les K-Mean ont encore quelques défauts, car la sortie du cluster dépend principalement de la sélection des centroïdes initiaux des clusters. De plus, le nombre de grappes k doit être donné à l'avance. en outre, les clusters résultants n'incluent pas toutes les possibilités des instances de classe. [43] a utilisé l'algorithme K-Mean pour regrouper et analyser les données. Ils ont utilisé l'apprentissage non-supervisée pour la détection d'intrusion. Leur approche méthode permet de détecter efficacement les intrusions inconnues dans les connexions réseau réelles.

4.9 La corrélation d'alertes

Pour protéger les infrastructures contre les attaques, plusieurs méthodes avaient été utilisées telles que les pare-feux et les systèmes de détection d'intrusion ainsi que les différents mécanismes de contrôle d'accès et d'authentification. Force est de constater que ces mécanismes ne sont pas totalement efficaces et génèrent beaucoup d'alertes qui ne présentent pas forcement des menaces. Il faut donc mettre en place une technique permettant de gérer efficacement les alertes tout en assurant la continuité du service. La corrélation d'alertes intervient afin d'apporter une solution aux problèmes causés par les systèmes de détection d'intrusion tels que les fausses alertes, la duplication des alertes, le manque de standardisation des alertes.

- **Les fausses alertes** : les fausses alertes sont des nombreuses alertes qui sont déclenchées par des IDS lorsqu'un comportement diffère légèrement des activités normales d'un utilisateur. Très souvent, ces alertes ne constituent pas toutes des menaces. Parfois, ce n'est que le contexte dans lequel se déroule l'activité qui détermine si elle est intrusive. En raison des exigences sévères en temps réel, les systèmes de détection d'intrusion ne peuvent pas analyser le contexte de toutes les activités dans la mesure requise.

- **La duplication des alertes** : pour pallier aux problèmes des fausses alertes générées de façon abusives, la corrélation d'alertes a pour but de réduire les fausses alertes afin de faciliter la tâche de l'analyste de sécurité. Les vraies alertes sont trop souvent incluses dans les fausses alertes. Elles ne peuvent pas être traitées ou tardivement traitées. Ce qui constitue un problème car en matière de sécurité informatique, face à un problème tel que la détection d'intrusion les décisions conséquentes en cas d'attaque doivent être prises à temps pour limiter les effets de l'attaque sur les infrastructures.

- **Le manque de standardisation des alertes** : face à la diversité des systèmes de détection d'intrusion, on constate que le format des alertes générées ne sont pas les mêmes d'un IDS à un autre. Il faut donc mettre en place un système qui permet d'agréger les alertes similaires sous un format standard.

Les systèmes classiques de corrélation d'alertes comportent 3 familles de méthodes de corrélation :

- La famille **probabiliste** ou *la corrélation implicite* : au sein de cette famille, les alertes sont corrélées sur base de la similarité de leurs caractéristiques. L'exploitation des alertes révèle des relations intrinsèques entre elles. Une relation est constituée par exemple par une correspondance fréquentielle ou statistique entre des alertes. Cette méthode ne

permet pas de retrouver les traces des alertes car elle ne fournit pas de relation causale entre les alertes.

- La famille **LAMBDA** ou *la corrélation explicite* : l'opérateur est capable d'exprimer explicitement des relations entre différentes alertes, sous la forme d'un scénario. Un scénario regroupe un ensemble de propriétés que doivent satisfaire les alertes, et des liens les connectant. Cette méthode est restreinte aux types d'attaques connues et si une attaque récente à été utilisée, cette méthode ne peut pas la détecter. Le langage LAMBDA est utilisé comme outil de représentation des scénarios d'attaques. Le problème avec cette approche implique qu'il faut un expert pour fournir les scénarios d'attaques aussi complet que possible.

- L'utilisation des **pré-conditions et post-conditions** ou **mixte** : L'utilisation des pré-conditions et post-conditions se base sur la description logique des attaques sous la forme d'un triplet en reprenant l'attaque et ses pré et post conditions. Les alertes sont donc corrélées lorsque les post-conditions d'une alerte correspondent aux pré-conditions d'une alerte qui est générée. *CRIM* (Coopération et reconnaissance d'Intention Malveillante) est un prototype qui fut développé pour offrir une base de travail. Cette méthode permet de découvrir les liens causaux entre les alertes et de fournir un diagnostic assez précis.

Plusieurs travaux ont été effectués avec succès en utilisant les différents algorithmes du Machine Learning. La corrélation d'alertes via le Machine Learning peut être vue comme étant un problème de type supervisé ou non-supervisé. En effet, tout dépend de la manière dont on aborde le problème de la corrélation d'alertes. Si le but consiste à regrouper les alertes similaires de manière automatique l'approche non-supervisée est la plus adéquate par contre, si on souhaite utiliser l'apport humain afin de valider ou rejeter les fausses alertes alors l'apport supervisé serait le bon choix.

De façon générale, la corrélation d'alertes via le Machine Learning comporte deux étapes [94] [73] à savoir : l'étape de la collection et de la normalisation ainsi que l'étape de la fusion ou d'agrégation. En ce qui concerne les sources de données, les données de DARPA [51] [73] sont souvent utilisées dans le cadre de la corrélation d'alertes.

La première étape est celle du prétraitement ainsi que le processus de normalisation des alertes et la seconde phase consiste à l'utilisation des algorithmes du Machine Learning afin de détecter des vraies alertes et des fausses alertes. Rappelons que la distinction des vraies alertes des fausses alertes constitue l'objectif principal de la corrélation d'alertes. La phase de normalisation et de pré-traitement permettent de distinguer les FP et de faire une analyse

profonde des alertes [94]. Les alertes générées peuvent être sauvegardées dans une base de données et doivent être standardisées lorsqu'elles proviennent de plusieurs IDS différents. Un exemple de standardisation des messages tels que "scanning", "nmap scan" appartiennent à la catégorie de "Scan" illustré par [94]. Notons que cette phase peut aider l'analyste de sécurité d'avoir une idée de la catégorie d'alertes. A l'issue de la normalisation et du prétraitement, il faut donner les labels (étiquettes) aux alertes puis regrouper les alertes ayant les mêmes caractéristiques ainsi qu'une petite intervalle de temps entre elles. Le temps de ces alertes sont donc fusionnés afin de faciliter la tâche de la généralisation.

La phase de généralisation des alertes incorpore un classement hiérarchique basé sur l'IP, le port et le temps. Une phase de vérification basée sur les informations récoltées sur les vulnérabilités des machines disponibles via **Nessus**[4] permet de récolter les vulnérabilités des machines disponibles. A l'issue de la vérification chaque alerte est étiquetée vraie positive ou fausse positive en fonction du résultat du Nessus sur les machines. L'utilisation d'un algorithme de classification **RIPPER**[5] permet de former le modèle afin de créer un modèle automatique de classification d'alertes sur base des règles.
Cette technique permet donc de supprimer les alertes redondantes générées par les systèmes de détection d'intrusion. Une comparaison a été réalisée entre diverses méthodes du Machine telles que *Decision Stump*[6], *OneR*[7], l'arbre aléatoire (Random Forest) via les mesures de performance a donné des résultats meilleurs.

[73] propose un système de corrélation d'alertes réseau en utilisant l'approche non-supervisée nécessitant aucune intervention humaine. Lors de la première étape, les alertes sont regroupées de sorte que chaque groupe constitue une étape d'une attaque. Lors de la deuxième étape, les groupes créés lors de la première étape sont combinés de telle sorte que chaque combinaison de groupes contient les alertes d'une seule attaque complète.

[4]Nesuss est un logiciel de scan des vulnérabilités

[5]Produce Error Reduction (RIPPER) est un algorithme basé sur des règles d'association avec une réduction de l'élagage des erreurs

[6]est un modèle du Machine Learning machine composé d'un arbre de décision à un niveau

[7]OneR ou "One Rule" est un algorithme de classification simple, mais précis, qui génère une règle pour chaque prédicteur dans les données, puis sélectionne la règle avec la plus petite erreur totale en tant que règle unique

Chapitre 5

La détection de spam via le Machine Learning

5.1 L'implémentation

Le système de détection de spams proposé s'inspire de [22]. Le but de l'implémentation est de montrer qu'en se basant sur ladite source, on peut arriver à améliorer la performance de la détection de spam en utilisant à bon escient les différentes techniques du Machine Learning.

5.1.1 Les datasets

Le travail se base sur un dataset public pour la phase d'entraînement **LingSpam** [17](L). Ce dataset contient au total 702 fichiers bruts d'emails dont la moitié est classée spam et l'autre moitié non-spam. Pour tester le modèle, un dataset fourni par [17] sera utilisé à cet effet. Ce dataset contient 260 fichiers d'emails dont la moitié est classée spam et l'autre classée non-spam, par soucis de clarté, le dataset de test sera noté dataset.

	Dataset L	Dataset T
Spam	351	130
Non-spam	351	130
Total	702	260

TABLE 5.1 – Les datasets L et T utilisés pour l'entraînement et le test.

5.1.2 La création du corpus

Après l'acquisition des données brutes des emails de spams et de non-spams pour la phase d'entraînement effectuée, il est important de créer un corpus c'est-à-dire un vecteur contenant tous les mots des emails. Ensuite, une phase de nettoyage a permis de supprimer tous les stop-words (ce sont les mots les plus fréquemment utilisés et qui ne rapportent pas plus d'informations voir Figure 5.1). Ce nettoyage a aussi pour but de gagner en performance en réduisant le nombre de caractéristiques non-indispensables.

a	about	above	after	again	against	all	am
an	and	any	are	aren't	as	at	be
because	been	before	being	below	between	both	but
by	can't	cannot	could	couldn't	did	didn't	do
does	doesn't	doing	don't	down	during	each	few
for	from	further	had	hadn't	has	hasn't	have
haven't	having	he	he'd	hell	he's	her	here
hers	here's	her's	her	herself	him	himself	his
how	how's	i	i'd	i'll	i'm	i've	if
in	into	is	isn't	it	it's	itself	let's
me	more	most	mustn't	my	myself	no	nor
not	of	off	on	once	only	or	other
ought	our	ours	ourselves	out	over	own	same
she	shan't	she'd	she'll	she's	should	shouldn't	so
some	such	than	that	that's	the	their	theirs
them	themselves	then	there	there's	these	they	they've
they're	they'll	they'd	this	those	through	to	too
under	until	up	very	was	wasn't	we	we've
were	we're	weren't	what	what's	when	when's	where
where's	which	while	who	who's	whom	why	why's
with	won't	would	wouldn't	you	you'd	you've	you'll
your	yours	yourself	yourselves				

FIGURE 5.1 – La liste des stops-words utilisés dans l'expérimentation.

5.1.3 L'extraction des caractéristiques

Nous avons sélectionné tous les termes du document (les fichiers de spams et non-spams) qui représentent les documents et les convertir en une dimension dans l'espace vectoriel. Ce

processus implique au préalable le nettoyage des mots d'arrêts (stop words) dans tous les documents car ils ne rapportent pas d'informations en utilisant la méthode **TF-IDF**[1].
La méthode **TF-IDF** correspond à la fréquence de fréquence inverse du document et le poids du TF-IDF. Elle est une méthode pour évaluer l'importance d'un mot dans un document. L'importance augmente proportionnellement au nombre de fois qu'un mot apparaît dans le document mais est compensé par la fréquence du mot dans le corpus. La fréquence du terme *t* (TF) mesure la fréquence à laquelle le terme *t* se produit dans un document se calcule de la manière suivante :

TF (*t*) = (le nombre de fois où le terme *t* apparaît dans un document) / (le nombre total de termes dans le document).

La partie IDF mesure l'importance d'un terme. En calculant TF, tous les termes sont considérés comme étant tout aussi importants. Cependant, il est connu que certains termes, comme les stop-words peuvent apparaître plusieurs fois mais ont peu d'importance. Ainsi, il faut peser les termes fréquents tout en augmentant les rares, en calculant de la manière suivante :
IDF (*t*) = log (le nombre total de documents / le nombre de documents avec le terme *t*).
L'importance ou le poids $W_{t,d}$ d'un terme *t* dans un document *d* se calcule comme suit :
$W_{t,d}$ = le nombre de fois où le terme *t* apparaît dans le document *d* x log (le nombre total de document / le nombre de documents avec le terme *t*).

5.1.4 La création et le test des modèles

La détection des spams dans la littérature est approchée par les algorithmes classification. Le but est de classer si un email est un spam ou non. L'algorithme de Naïve Bayes, l'arbre de décision, le SVM , la régression logistique seront utilisés dans la phase d'entraînement.

5.2 Évaluation

L'expérimentation du Naïve Bayes sur le dataset **LingSpam** [17](L) via le *10-fold cross-validation* a permis d'obtenir le résultat suivant :

[1] Frenquence-Invers Document Frenquency

K	1	2	3	4	5	6	7	8	9	10
Résultat	1	1	1	0.98571429	0.97142857	0.94285714	0.95714286	0.98571429	1	0.98571429

TABLE 5.2 – Le résultat du 10-fold cross-validation sur les modèles avec le dataset L via le Naïve Bayes.

L'idée principale de l'utilisation du 10-fold cross-validation est d'estimer la performance du modèle sur les mêmes données qui ont été utilisées afin de créer le modèle. On constate que le modèle parvient à généraliser avec de très bon score sur les données deja utilisées dans la phase de la formation du modèle.

5.3 Analyse de l'implémentation

Le choix porté sur le dataset L est fait à l'issu d'un test de comparaison avec un dataset **Enron** [100] (E). Le dataset E contient 5857 fichiers d'emails dont 1496 fichiers d'emails classés spams. Pour choisir le dataset qui permet de créer un modèle permettant de détecter et classifier les spams, pour chaque dataset, un modèle de classification de spam fut conçu, et tous les deux datasets(L et E) ont été testés avec le dataset de test T. La matrice de confusion est l'outil qui permettra de comparer les résultats de chaque modèle. Le résultat du test du modèle (conçu avec le dataset E en utilisant le Naïve Bayes) en testant avec le dataset T donné par la matrice de confusion la Figure 5.2.

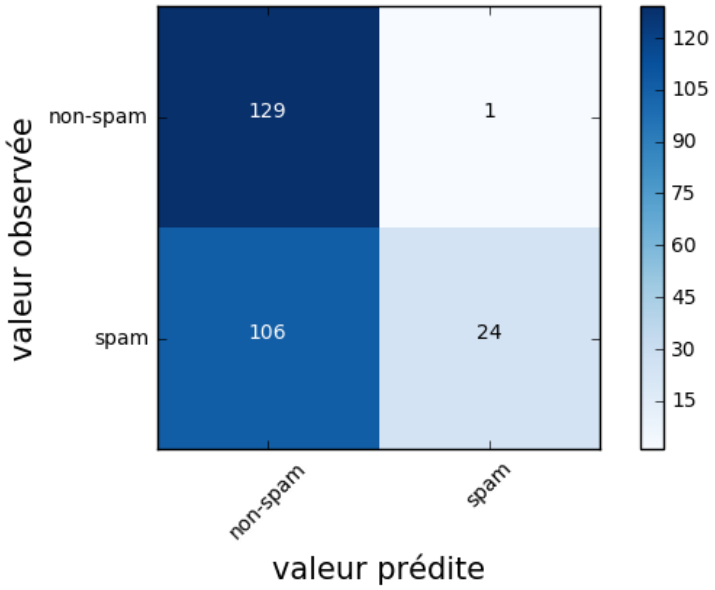

FIGURE 5.2 – La matrice de confusion du Naïve Bayes sur le dataset E.

L'expérimentation du Naïve Bayes sur le dataset **Enron** [100] en utilisant la mesure ROC est donnée par la figure 5.3

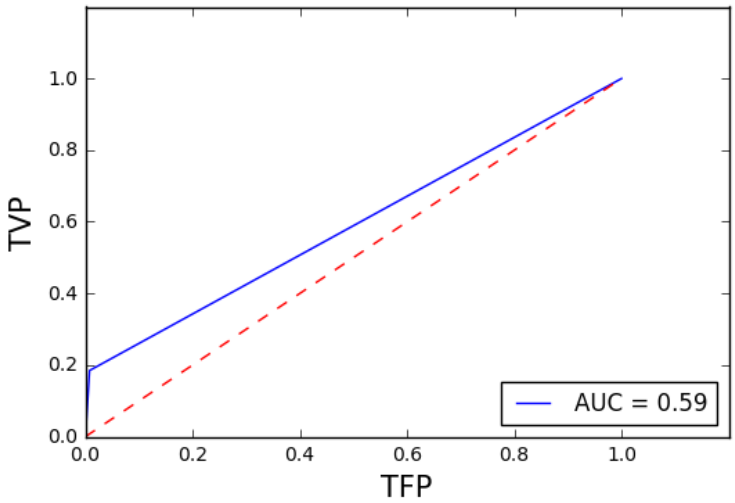

FIGURE 5.3 – La courbe de ROC du Naïve Bayes sur le dataset E.

On constate que le modèle construit par le Naïve Bayes sur le dataset E génère beaucoup de faux négatifs(106) et ce dataset n'est donc pas très efficace pour construire un véritable modèle.

Dans les écrits, plusieurs approches consistent à sélectionner les tops **n** termes ayant les plus fortes fréquences d'apparition dans les emails spams et non-spams. A ce propos, le choix portera sur les tops **2000** premiers termes. La raison principale du choix du mot est qu'après plusieurs tests, le nombre idéal de termes (caractéristiques) se situe autours de 2000 termes car le nombre de faux positifs et faux négatifs sont les plus petits.
En opposant la technique utilisée dans le cadre de l'implémentation avec la sélection des tops **2000** caractéristiques, plusieurs remarques peuvent être faites :

- **Le temps** : La technique de la sélection des tops **2000** caractéristiques est plus lente et plus couteuse en terme de temps comparée à la technique d'extraction **TF-TDF** utilisée dans l'implémentation.
- **La mesure de performance** : Avec la technique de sélection des **tops 2000**, on constate une légère amélioration. Cependant le nombre des faux négatifs (9) peut être encore amélioré (Figure 5.4).

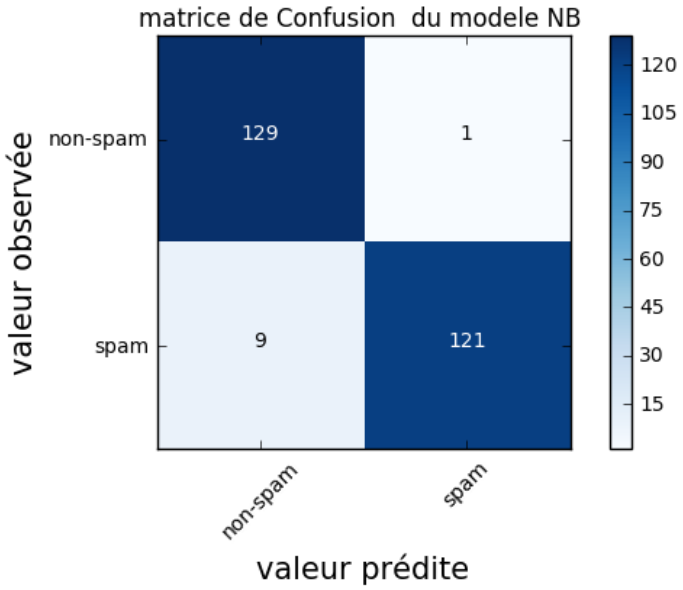

FIGURE 5.4 – La matrice de confusion du Naïve Bayes sur le dataset L avec la méthode de sélection des Tops **2000** caractéristiques.

La mesure de l'aire sous la courbe permet d'illustrer le résultat de l'implémentation en utilisant la méthode de sélection des 2000 meilleures caractéristiques (Figure 5.5).

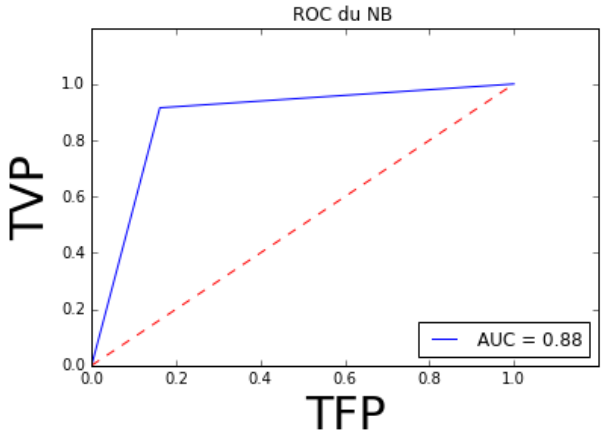

FIGURE 5.5 – La courbe ROC du Naïve Bayes sur le dataset L avec la méthode de la sélection des Tops **2000** caractéristiques.

	Precision	Recall	F1-score
Non-spam	0.91	0.84	0.87
Spam	0.85	0.92	0.88
Avg	0.88	0.88	0.88

TABLE 5.3 – Les mesures de performances du Naïve Bayes sur le dataset L avec la méthode de la sélection des Tops **2000** caractéristiques .

5.3.1 Le choix des classifiers

Dans cette section, nous allons discuter du choix de l'algorithme utilisé. Nous essayerons de comparer l'algorithme de Naïve Bayes avec l'arbre de décision, la régression logistique, le Support Vector Machine ainsi que l'arbre de décision et d'analyser les résultats obtenus. La comparaison et l'analyse des résultats seront fait via des outils de mesures de performances suivantes : **la matrice de confusion, la courbe ROC, la précision, le rappel, le f1-score**. A la fin nous allons utiliser les techniques permettant d'optimiser les paramètres afin d'améliorer les résultats obtenus.

L'arbre de décision utilisée avec les paramètres par défaut a permis d'obtenir les résultats pas très satisfaisants. Le but de ce choix est de faire un premier essaie de solution dans la démarche de la recherche d'un meilleur résultat. Les paramètres de l'arbre de décision utilisés sont : min_samples_leaf=1, min_samples_split=2,criterion='gini', max_depth=None. A travers ces paramètres nous pouvons constater que le nombre minimal d'échantillons requis pour diviser un noeud interne est de 2 et ainsi que le nombre minimum d'échantillons requis

pour être à un nœud de feuille est de 1. Nous pouvons également constater que la profondeur de l'arbre est maximale. Les noeuds sont développés jusqu'à ce que toutes les feuilles soient pures ou jusqu'à ce que toutes les feuilles contiennent moins d'échantillons spécifiés par la valeur du min_samples_split.
Le résultat avec les configurations par défaut de l'arbre de décision ne sont pas satisfaisants mais on constate qu'il y'a encore un nombre élevé des faux négatifs(9) et faux positifs(18) (Figure 5.6) qui peut être encore amélioré.

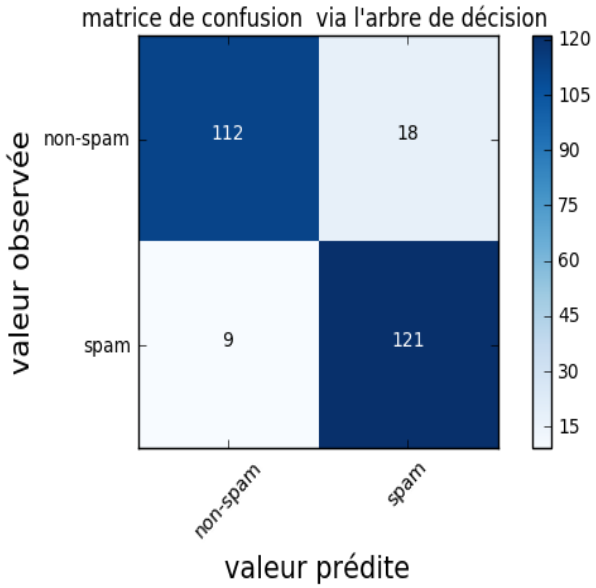

FIGURE 5.6 – La matrice de confusion de l'arbre de décision avec les paramètres par défaut.

Cela se confirme à travers la valeur de la courbe du ROC calculée via l'aire à 0.90 (Figure 5.7).

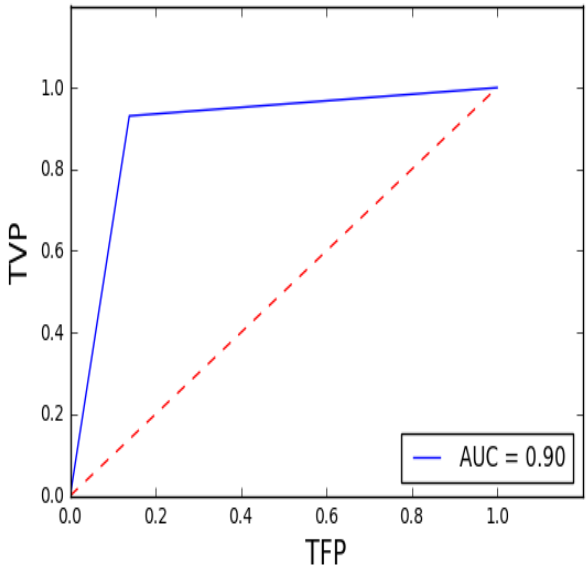

FIGURE 5.7 – La courbe ROC de l'arbre de décision avec les paramètres par défaut.

Le paramètre min_samples_leaf à travers la figure nous permet de fixer le nombre minimum d'échantillons requis pour être au niveau d'un noeud feuille. En modifiant les paramètres (criterion = "gini", min_samples_leaf=5), nous obtenons une petite amélioration comparée au résultat en utilisant les paramètres par défaut de l'arbre de décision.

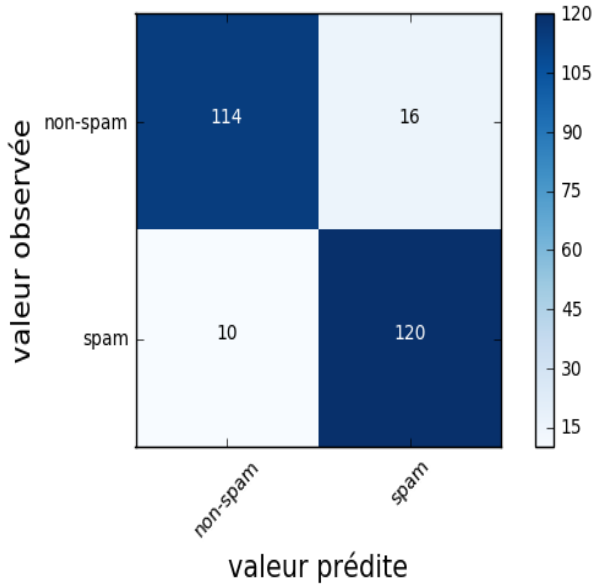

FIGURE 5.8 – La matrice de confusion de l'arbre de décision via le GINI.

Ainsi que la courbe de ROC donnée par la figure 5.13.

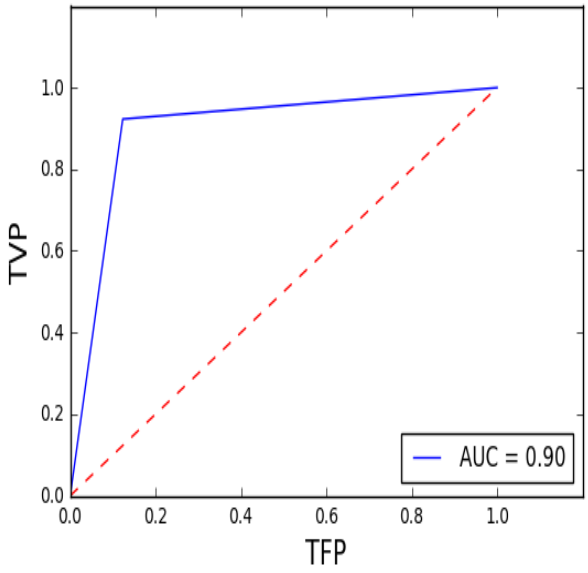

FIGURE 5.9 – La courbe ROC de l'arbre de décision via le GINI.

En utilisant le critère d'**entropie** et fixant le paramètre **min_samples_leaf** à 6, on constate une légère augmentation de la détection des vrais négatifs et une diminution des faux positifs. Le paramètre **min_samples_leaf** fixé à 6, après plusieurs tests on constate que la valeur 6 a permis d'avoir un meilleur résultat(0.907692307692) que les autres (Figure 5.10).

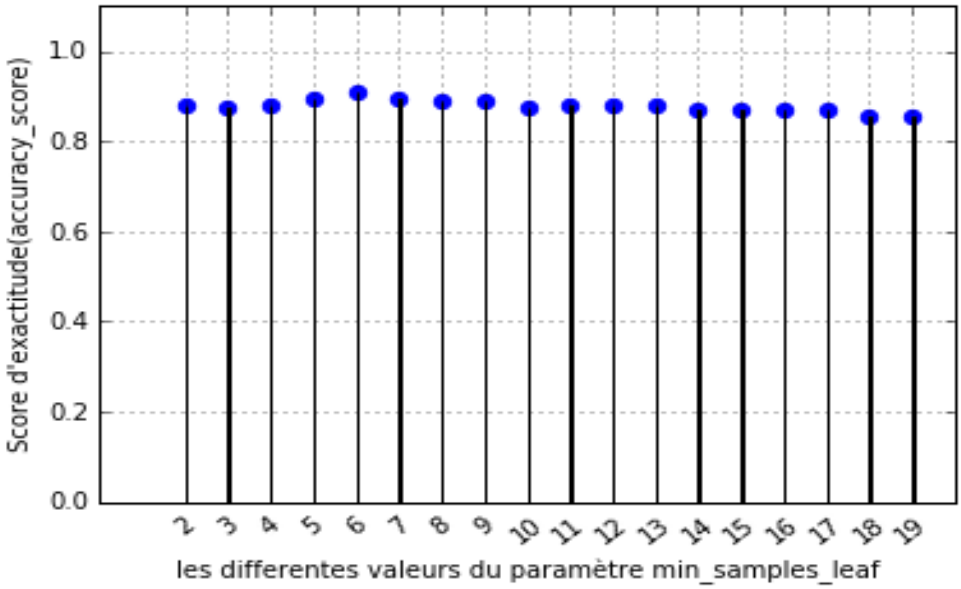

FIGURE 5.10 – Les différents des résultats d'exactitude du paramètre min_samples_leaf de 2 à 19

On constate que plus le score se détériore au fur et à mesure que la valeur de **min_samples_leaf** augmente (Figure 5.11)

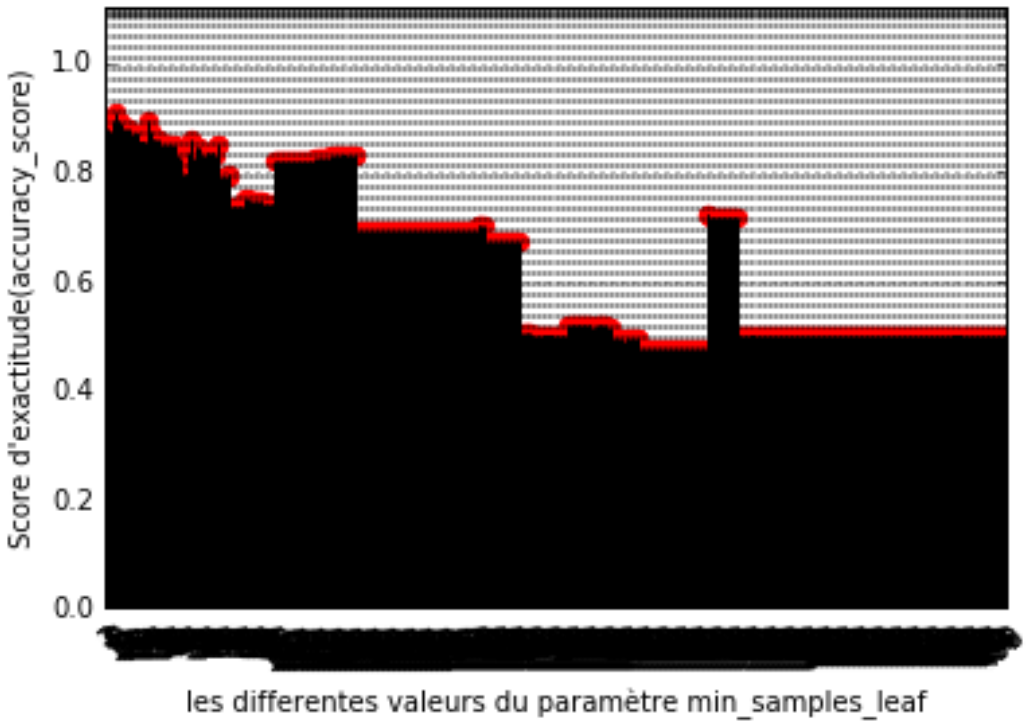

FIGURE 5.11 – Les différents résultats d'exactitude du paramètre min_samples_leaf de 2 à 500.

Avec le paramètre **min_samples_leaf** fixé à 6, le modèle construit a permis d'améliorer très légèrement sa performance comme le montre la matrice de confusion (Figure 5.12). Un gain de 1 % de l'aire sous la courbe (Figure 5.13) a été observé.

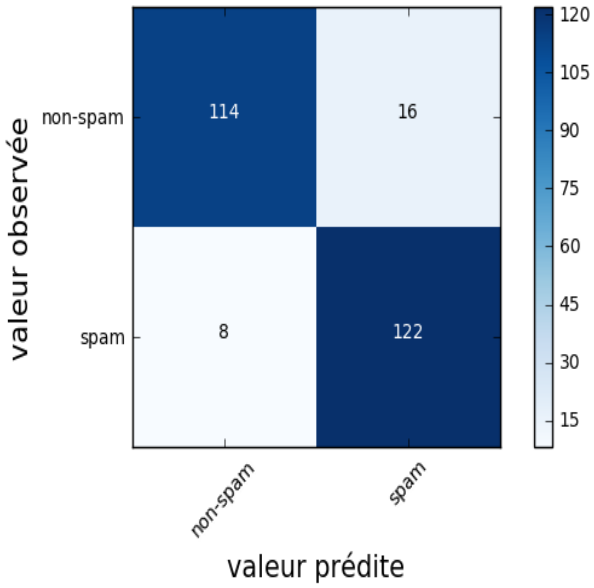

FIGURE 5.12 – La matrice de confusion de l'arbre de décision via l'entropie.

Ainsi que la courbe ROC :

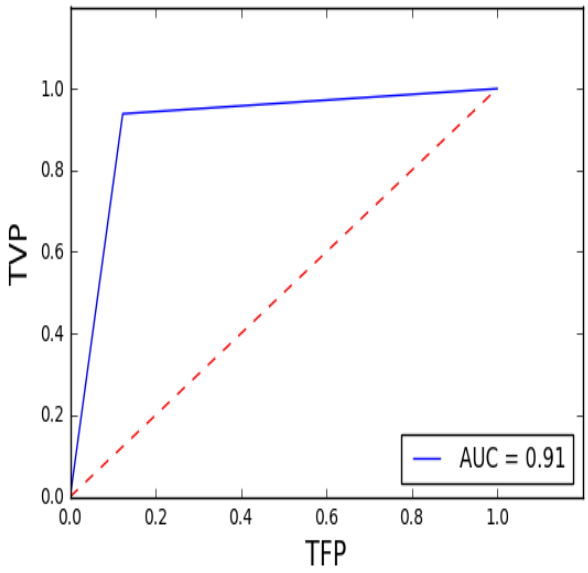

FIGURE 5.13 – La courbe ROC de l'arbre de décision via l'entropie.

En utilisant la régression logistique, le modèle a donné des résultats très significatifs. En effet, les précisions de la détection des spams et des non-spams ont atteint respectivement 98% et 97%. La matrice de confusion permet d'illustrer le résultat de l'experience (Figure 5.14).

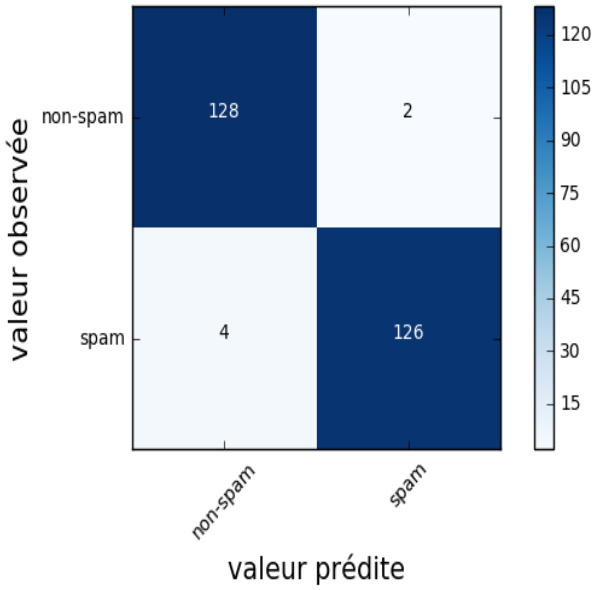

FIGURE 5.14 – Le résultat de la matrice de confusion via la régression logistique.

La courbe ROC permet également de visualiser l'étendu de la détection. L'aire sous la courbe obtenu via la figure 5.15 a atteint un bon résultat de 98 %.

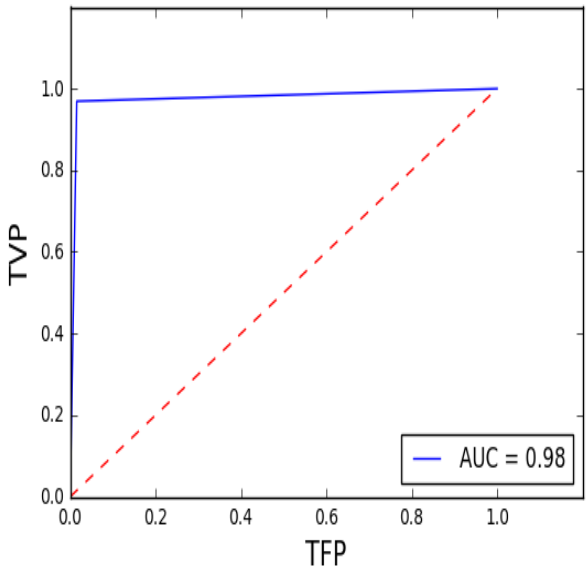

FIGURE 5.15 – La courbe de ROC via la régression logistique.

Le récapitulatif des mesures de performance permet de voir le résultat des autres mesures de performances telles que la Précision, le Recall et la mesure F1.

	Précision	Recall	F1-score
Non-spam	0.97	0.98	0.98
Spam	0.98	0.97	0.98
Avg	0.98	0.98	0.98

TABLE 5.4 – Récapitulatif des mesures de performances en utilisant la régression logistique.

En utilisant le SVM, le modèle a donné un bon résultat dans le cadre de la détection des spams. Le modèle a été conçu avec une faible valeur du gamma fixé à 0.004. Ce résultat est illustré par la matrice de confusion (Figure 5.16).

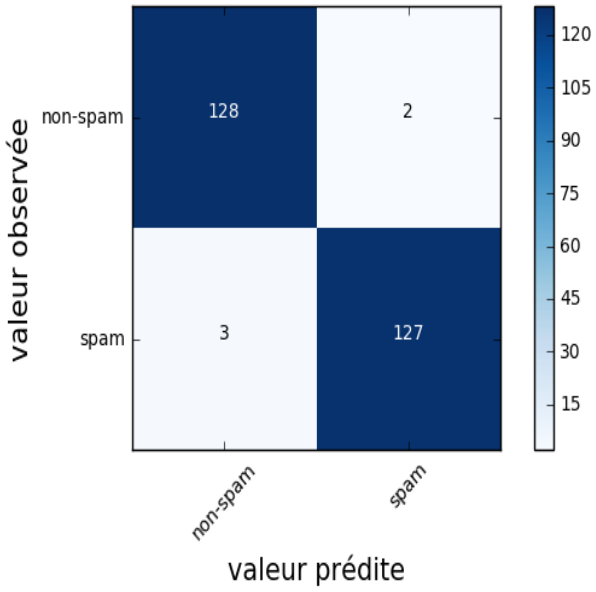

FIGURE 5.16 – Résultat de la matrice de confusion via le Support Vector Machine.

La courbe ROC permet également de visualiser l'étendu de la détection. L'aire sous la courbe obtenu via la figure 5.17 a atteint un bon résultat de 98 %.

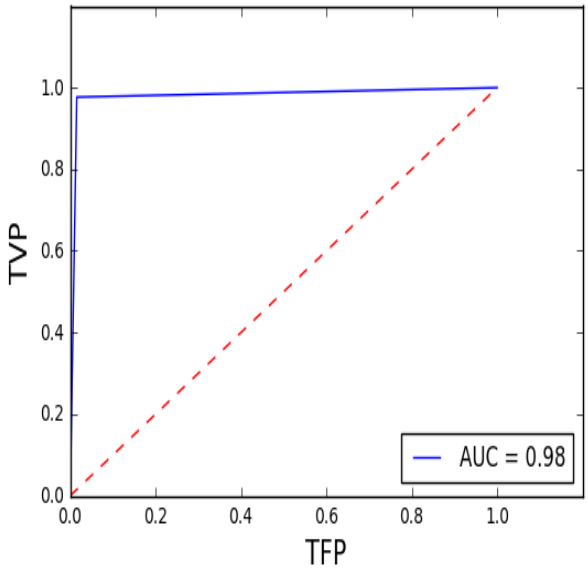

FIGURE 5.17 – La courbe de ROC via le Support Vector Machine.

En effet, le paramètre gamma définit à quelle distance l'influence d'un seul exemple de formation atteint, avec des valeurs faibles signifiant «loin» et des valeurs élevées signifiant «proches». On remarque également qu'en augmentant le paramètre gamma le résultat est affecté négativement montrée par figure 5.18.

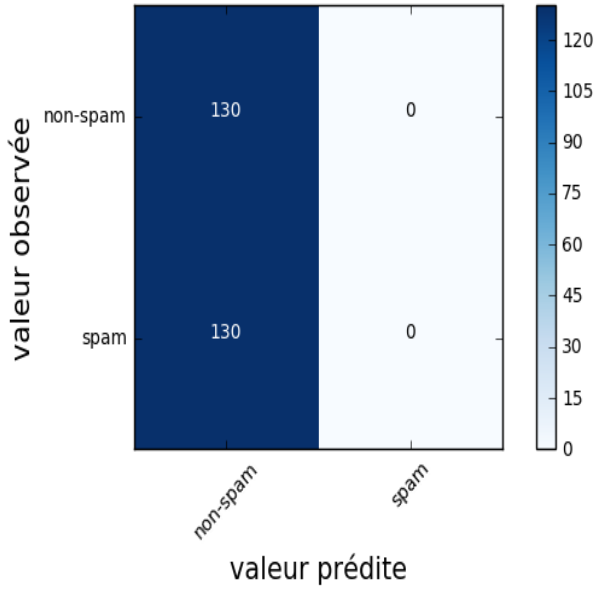

FIGURE 5.18 – Résultat de la matrice de confusion via le Support Vector Machine avec gamma à 1000.

Pour justifier l'argumentation, la table 5.5 montre un récapitulatif des résultats faibles en augmentant la valeur du paramètre Gamma.

	precision	recall	f1-score
non-spam	0.50	1.00	0.67
spam	0.00	0.00	0.00
avg / total	0.25	0.50	0.33

TABLE 5.5 – Récapitulatif des mesures de performance en utilisant un gamma élevé via le Support Vector Machine .

Le paramètre C comme nous l'avons vu dans les autres chapitres permet d'anticiper les erreurs de classifications. Dans le cadre de l'implémentation, nous avons testé la valeur de C. A partir de 5, la valeur de C reste constante à 0.980769230769. La figure 5.19 permet de constater les résultats en variant les différentes valeurs de C.

TABLE 5.6 – La détermination de la bonne valeur de C de 1 à 10

C	1	2	3	4	5
Résultat	0.957692307692	0.976923076923	0.976923076923	0.976923076923	0.980769230769

Et de 6 à 10.

TABLE 5.7 – La détermination de la bonne valeur de C de 6 à 10

C	6	7	8	9	10
Résultat	0.980769230769	0.980769230769	0.980769230769	0.980769230769	0.980769230769

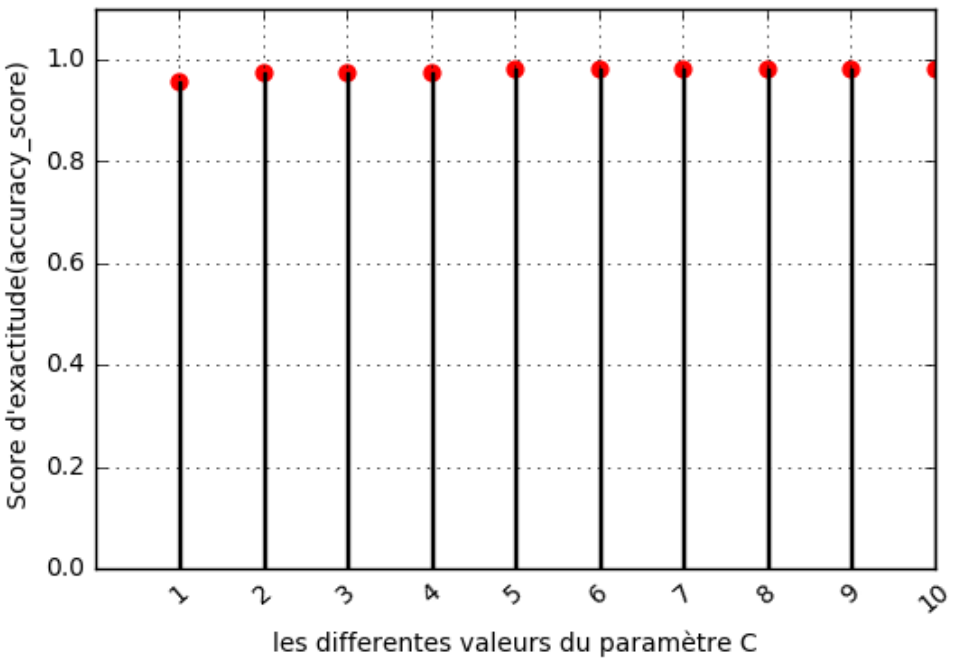

FIGURE 5.19 – Visualisation de la détermination de la valeur de C via le Support Vector Machine.

La détection de spam en utilisant le support Vector Machine a permis d'atteindre un bon niveau de précision en diminuant la valeur du paramètre gamma (Figure 5.8). Globalement, les mesures de précision ont donné des résultats relativement satisfaisants.

	precision	recall	f1-score
non-spam	0.98	0.98	0.98
spam	0.98	0.98	0.98
avg / total	0.98	0.98	0.98

TABLE 5.8 – Récapitulatif des mesures de performances en utilisant le Support Vector Machine .

5.4 Proposition d'amélioration du Naïve Bayes

Dans cette section, nous allons tenter de proposer une stratégie afin d'améliorer les résultats des différents algorithmes utilisés. Cette stratégie passe par la détermination des bons paramètres pour les algorithmes.

La remarque qui peut être faite à l'utilisation du Naïve Bayes dans l'implémentation précédente concerne la determination des meilleurs paramètres. La difficulté principale de l'utilisation d'un algorithme du Machine Learning réside dans la determination des paramètres adéquats afin d'obtenir des résultats satisfaisants. En effet, le résultat peut être amélioré en modifiant les paramètres ce qui constitue le coeur de cette section.
Afin d'améliorer le résultat, nous allons construire le modèle en utilisant la méthode **GridSearchCV**. La méthode GridSearchCV permet de déterminer les meilleures hyper-paramètres du modèle et de proposer un modèle permettant de détecter efficacement les spams.

Rappelons que l'hyper-paramètre est une configuration externe au modèle et dont la valeur ne peut pas être estimée à partir des données. Les hyper-paramètres servent à optimiser les performances du modèle.
Nous avons utilisé le **Pipeline**[2] afin de combiner de manière séquentielle l'exécution des actions suivantes : extraire les ensembles de caractéristiques, compter les fréquences via **CountVectorizer**, appliquer le TF-IDF et ensuite utiliser l'algorithme Naïve Bayes.

L'utilisation de la méthode GridSearchCV dans le cadre de la recherche des hyper-paramètres nécessite la fixation des paramètres tels que la fréquence maximum du vecteur de fréquence ('vect_max_df' : (0.9, 0.98))), l'intervalle d'utilisation du n-gram ('vect_ngram_range' : [(1,2),(1,3)]), le paramètre alpha du modèle ('clf_alpha' : ($1e-1$, $1e-2$)). La figure 5.9 montre les meilleurs hyper-paramètres sélectionnés sur base du score.

Paramètres	Valeurs
clf_alpha	0.01
vect_max	0.9
vect_ngram	(1,2), (1,3)
Meilleur score	0.984

TABLE 5.9 – Le résultat de la recherche des meilleurs hyper-paramètres.

Le résultat du test avec le dataset T a permis d'obtenir le résultat relativement meilleur que les autres résultats (Figure 5.2) illustrés par la matrice de confusion(Figure 5.20).

[2]Pipeline class est un outil pour encapsuler plusieurs transformateurs différents aux côtés d'un estimateur en un objet

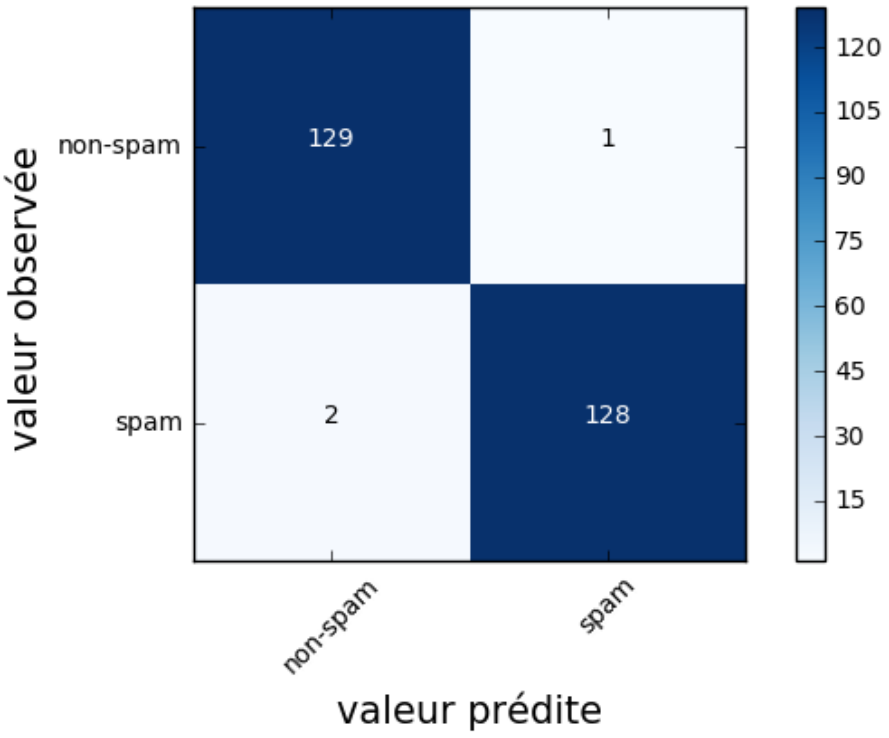

FIGURE 5.20 – La matrice de confusion via les meilleurs paramètres.

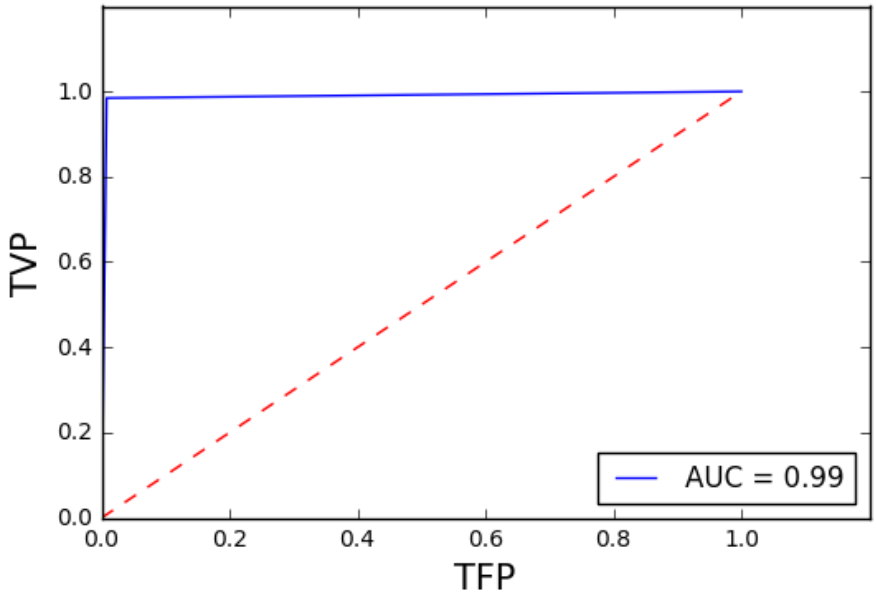

FIGURE 5.21 – La courbe ROC en utilisant GridSearchCV.

Un récapitulatif des mesures de performance nous permet de constater que le résultat de

chaque mesure de performance se rapproche proche de 1, ce qui constitue une amélioration (Figure · 5.10).

	precision	recall	f1-score
non-spam	0.98	0.99	0.99
spam	0.99	0.98	0.99
avg	0.99	0.99	0.99

TABLE 5.10 – Le récapitulatif des mesures des performances.

Chapitre 6

Conclusion

A travers ce travail, nous avons fait l'état de l'art du Machine Learning dans son ensemble en expliquant les étapes de son application face à un problème de sécurité. Nous avons également discuté des différents composantes du Machine Learning ainsi que des mesures de performances permettant de porter des jugements sur la qualité de l'apprentissage.

Dans l'état de l'art, le Machine Learning a permis de faire l'identification à travers l'empreinte digitale dans le cadre de la biométrie. L'identification par empreinte digitale a permis d'apporter des solutions aux problèmes des mot de passe. Les algorithmes tels que le SVM et les réseaux de neurones qui fonctionnent par détection des patterns dans les données ont permis de déceler des modèles de reconnaissance des empreintes digitale à travers les caractéristiques relatives aux empreintes digitales. Un aspect dynamique de la biométrie à savoir l'authentification par Keystroke a été explorée par le Machine Learning. Le Keystroke renforce le caractère statique de l'authentification via les mot de passe. Le Machine Learning à travers les algorithmes utilisés dans les travaux a permis d'authentifier des utilisateurs sur base de leurs caractéristiques d'interaction avec les claviers.

Le Machine Learning a également contribué à la détection des botnets qui s'infiltrent dans les ordinateurs et se propagent sur tous les systèmes informatiques à travers le réseau. Cette détection se fait par détection des signatures des bots existantes ou par détection des comportements anormaux (anomalie). Plusieurs travaux ont été mené afin de lutter contre les botnets à travers les algorithmes tels que le SVM, le KNN , les arbres de décisions.

Le Machine Learning a permis de detecter les tunnels DNS qui permettent le vol des informations personnelles à des mauvaises fins à travers des protocoles de confiance telles que le HTTPS. Ces protocoles de confiances deviendront ainsi des canaux de communication permettant l'acheminement des données vers d'autres destinations. Ce détournement peut

se faire en utilisant les DGA utilisés fréquemment par des botnets. Le Machine Learning se base ainsi sur informations relatives aux noms DNS générés.

Les URLs malsaines sont très souvent des moyens utilisés par les hackers afin de piéger les utilisateurs et aussi charger les logiciels malveillants sur leurs postes. Les algorithmes du Machine Learning ont été utilisé afin de lutter contre ces URLs en se basant sur les caractéristiques des URLs telles que la longueur des URLs, la présence des adresses IP dans l'URL ainsi que le nombre de point au sein de l'URL. Ces informations ont permis de desceller les URLs malveillantes en utilisant les algorithmes de classification tels que le SVM, le Naïve Bayes ainsi que les arbres de décisions.

Le Machine Learning a contribué à detecter des injections SQL qui sont des sources de menaces sur des sites internets. Plusieurs travaux ont permis de detecter les injections SQL en utilisant le SVM, le Naïve Bayes ainsi que l'arbre de décision.

La détection des intrusions est un champ très largement exploré par le Machine Learning. La détection des botnets constitue une illustration de l'utilisation du Machine Learning dans la détection d'intrusion. De manière générale, le Machine Learning a contribué à detecter les intrusions sur base des signatures ou sur base d'un comportement anormale (par rapport à une base de comportements normaux connus). Dans les travaux, par exemple le K-MEAN a permis de regrouper les données afin de trouver les structures pouvant classer les structures comme anormales ou saines.

Les alertes capturées par les IDS peuvent être corrélées et fusionnées en trouvant les similitudes dans les alertes. Dans les travaux, le Machine Learning a permis de trouver des corrélations dans les alertes capturées par des IDS. Les résultats ont montré que le Machine Learning a réduit considérablement le nombre d'alertes à traiter.

Dans ce travail, nous avons analysé les problèmes de sécurité liés à la détection de spams et ainsi proposé une solution afin d'améliorer la performance à travers la recherche des hyper-paramètres. L'utilisation du Machine Learning nécessite le bon choix des paramètres à utiliser. En effet, nous avons montré dans ce document la mauvaise fixation des paramètres du SVM tel que le gamma et le paramètre C qui influencent négativement le résultat de la prédiction des spams. Nous avons également montré à travers ce document l'interêt de l'utilisation de la validation croisée combiné avec le GridSearchCV a permis de trouver les meilleurs paramètres afin d'obtenir des résultats améliorés. L'amélioration proposée a permis d'obtenir une précision de 99% ainsi que la valeur de la mesure du rappel de 99% .

Le Machine Learning dans le domaine de la cybersécurité via l'état de l'art effectué a montré des résultats satisfaisants. Cependant une remarque concernant l'authentification par Keystroke abordée par la classification peut être formulée. En effet, l'utilisation des algorithmes de classification dans le cadre de l'authentification par Keystroke suppose l'acquisition des données Keystroke des usurpateurs et des utilisateurs légitimes. Cependant dans un cadre réel d'utilisation, comment peux-t-on avoir les données des usurpateurs ?

Nous avons vue que le Machine Learning a contribué à résoudre des problématiques liées à la sécurité. Cependant comment pourrions-nous garantir la sécurité des modèles proposés par le Machine dans le domaine de la sécurité ? Autrement dit, que peut-on mettre en place pour assurer la sécurité du Machine Learning appliqué aux problèmes de sécurité ?

Bibliographie

[1] Ahmed Almusawi, Haleh Amintoosi, *DNS Tunneling Detection Method Based on Multilabel Support Vector Machine*, Computer Engineering Department, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

[2] AISSAOUI Sihem, Un système de détection d'intrusion basé sur les Séparateurs à Vaste Marge (SVM)

[3] Alexa Internet, Inc. 1996 - 2018, `https://www.alexa.com` ,10-March-2018

[4] Ali Shafigh Aski, Navid Khalilzadeh Sourati, *Proposed efficient algorithm to filter spam using machine learning techniques*, "Pacific Science Review A : Natural Science and Engineering", Volume 18, July 2016, Pages 145-149

[5] Ammar Yahya Daeef, R. Badlishah Ahmad, Yasmin Yacob, Mohd. Nazri Bin Mohd, *Lightweight Phishing URLs Detection Using N-gram Features*, "International Journal of Engineering and Technology (IJET)"

[6] Andres Munoz, Courant, "Machine Learning and Optimization ", Institute of Mathematical Sciences, New York, NY

[7] Aniruddh Ladole, D. A. Phalke, *SQL Injection Attack and User Behavior Detection by Using Query Tree, Fisher Score and SVM Classification*, "International Research Journal of Engineering and Technology", June-2016

[8] Ankita Bhaiyya, Prof. Sonali Bodkhe, Prof. Amit Pimpalkar, "International Journal of Computer Science and Mobile Computing", *Botnet Identification System Using Clustering and Machine Learning C5.0*, April-2015

[9] Anton Schwaighofer, *Sorting it out : Machine learning and fingerprints*, May 2002

[10] Antony Jeyanna J., E.Indumathi, D.Shalini Punithavathani, "A Network Intrusion Detection System Using Clustering and Outlier Detection", *International Journal of Innovative Research in Computer and Communication Engineering*, February-2015

[11] Badr HSSINA, Abdelkarim MERBOUHA, Hanane EZZIKOURI, Mohammed ERRITALI, *A comparative study of decision tree ID3 and C4.5*, "International Journal of Advanced Computer Science and Applications, Special Issue on Advances in Vehicular Ad Hoc Networking and Applications"

[12] Bin Yu, Daniel L. Gray, Jie Pan, Martine De Cock, and Anderson C. A. Nascimento, *Inline DGA Detection with Deep Networks*, 2017

[13] Bleha, S.A., Slivinsky, C., and Hussein,*Computer-Access Security Systems Using Keystroke Dynamics*, "IEEE Transactions on Pattern Analysis and Machine Intelligence",12, No. 12, pp. 12171222.

[14] Bo Dong, Xue Wang, "Comparison Deep Learning Method to Traditional Methods using for Network Intrusion Detection", IEE International Conference on Communication Software and Network, 8th, 2016

[15] Chris Sinclair, Lyn Pierce, Sara Matzner, "An Application of Machine Learning to Network Intrusion Detection", *Applied Research Laboratories The University of Texas at Austin*

[16] Christopher Bare, Support Vector Machines, `http://digitheadslabnotebook.blogspot.be/2011/11/support-vector-machines.html`, 30-November-2011, 23-August-2017

[17] Csmining, Ling-Spam data set, `http://csmining.org/index.php/ling-spam-datasets.html`, 2010, 16-August-2017

[18] Cuteprogramming, `https://cuteprogramming.wordpress.com/category/event/`, 8-November-2016, 21-August-2017

[19] Dinh-Tu Truong, Guang Cheng, *Detecting domain-flux botnet based on DNS traffic features in managed network*, "SECURITY AND COMMUNICATION NETWORKS"

[20] Doyen Sahoo, Chenghao Liu, Steven C.H. Hoi, *Malicious URL Detection using Machine Learning :A Survey*

[21] E-payment & Biometrics, GREYC-KeyStroke Dataset, `http://www.epaymentbiometrics.ensicaen.fr/greyc-keystroke-dataset`, 23-August-2017

[22] Email Spam Filtering : An Implementation with Python and Scikit-learn, http://www.kdnuggets.com/2017/03/email-spam-filtering-an-implementation-with-python-and-scikit-learn.html, March 2017, August 2017

[23] Erik Hellstrom, *Feature learning with deep neural networks for keystroke biometrics*, "Computer Science and Engineering, master's level", 2017

[24] Eun Hong Cheon, Zhongyue Huang, Yon Sik Lee, *Preventing SQL Injection Attack Based on Machine Learning*, "Department of Computer Information Engineering"

[25] Fabien Allard, Renaud Dubois, Paul Gompel, and Mathieu More, *Tunneling Activities Detection Using Machine Learning Techniques*, Thales Communications, Colombes Cedex, France

[26] Feng Zeng, Shuo Chang, Xiaochuan Wan, *Classification for DGA-Based Malicious Domain Names with Deep Learning Architectures*, "International Journal of Intelligent Information Systems"

[27] Fingerprint Identification using SVM, Team member : Xuan Xu, SUNetID : xuanx

[28] Frank Vanhoenshoven, Gonzalo Napoles, Rafael Falcon, *Detecting malicious URLs using machine learning techniques*

[29] Frédéric Majorczyk, Détection d'intrusions comportementale par diversification de COTS : application au cas des serveurs web. Informatique [cs]. Université Rennes 1, 2008.

[30] Gérald PETITJEAN, "INTRODUCTION AUX RESEAUX DE NEURONES"

[31] H. Delacour, A. Servonnet, A. Perro, J.F. Vigezzi and J.M. Ramirez, La courbe ROC (receiver operating characteristic) : principes et principales applications en biologie clinique, Revue general abc, 2005

[32] HALL, M.A AND SMITH L.A Feature subset selection : a correlation based filter approach.In *Proc of the International Conference on Neural Information Processing and Intelligent Information Systems* , Springer, pages 855-858, 1997

[33] Hamsa A. Abdullah, *Fingerprint Identification System Using Neural Networks*, Nahrain University, College of Engineering Journal (NUCEJ), Vol.15 No.2, 2012 pp234 - 244

[34] Hongliang Liu, Yuriy Yuzifovich, A DEATH MATCH OF DOMAIN GENERATION ALGORITHMS, https://blogs.akamai.com/2018/01/a-death-match-of-domain-generation-algorithms.html, 9-January-2018, 10-April-2018

[35] Howard Hamilton, Confusion Matrix, http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html, 8-June-2012, 15-August-2017

[36] https://web.stanford.edu/class/psych252/tutorials/Tutorial_LogisticRegression.html

[37] Isabelle Guyon & Andre Elisseeff, An Introduction to Variable and Feature Selection, Guyon et Elisseeff, Journal of Machine Learning Research 3, pages 1157-1182, 2003

[38] ISIR, Reconnaissance automatique des empreintes digitales, http://www.isir.upmc.fr/UserFiles/File/LPrevost/Biometrie%20Empreintes.pdf, 23-August-2017

[39] Issa Khalil, Ting Yu and Bei, Guan, Discovering Malicious Domains through Passive DNS Data Graph Analysis, June-2016

[40] Jason Brownlee, Supervised and Unsupervised Machine Learning Algorithms, http://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/, 2017, 16-August-2017

[41] Jason Brownlee, What is a Confusion Matrix in Machine Learning, http://machinelearningmastery.com/confusion-matrix-machine-learning/, 18-November-2016, 16-August-2017

[42] Jayveer Singh, LanishaJ.Nene, "A Survey on Machine Learning Techniques for Intrusion Detection Systems", *International Journal of Advanced Research in Computer and Communication Engineering, November 2013*,

[43] Jianliang Meng, Haikun Shang, Ling Bian, "The Application on Intrusion Detection Based on K-means Cluster Algorithm", *Information Technology and Applications*

[44] Jonathan Woodbridge, Hyrum S. Anderson, Anjum Ahuja, Daniel Grant, *Predicting Domain Generation Algorithms with Long Short-Term Memory Networks*, Endgame, Inc. Arlington, VA 22201

[45] Jonathan Woodbridge, Hyrum Anderson, Using Deep Learning To Detect DGAs, https://www.endgame.com/blog/technical-blog/using-deep-learning-detect-dgas, 18-NOVEMBER-2016, 01-April-2018

[46] Kevin Ross, Melody Moh, Teng-Sheng Moh, Jason Yao, *Poster : Multi-source data analysis for SQL injection detection*

[47] Keystroke Pressure-Based Typing Biometrics Authentication System Using Support Vector Machines

Wahyudi Martono, Hasimah Ali, and Momoh Jimoh E. Salami Intelligent Mechatronics System Research Group, Department of Mechatronics Engineering, International Islamic University Malaysia, PO BOX 10, 50728, Kuala Lumpur, Malaysia

[48] Kleber Stroeh, Edmundo Roberto Mauro Madeira and Siome Klein Goldenstein, "An approach to the correlation of security events based on machine learning techniques", *Stroeh et al. Journal of Internet Services and Applications, 4 :7*, 2013

[49] La Correlation, `http://grasland.script.univ-paris-diderot.fr/STAT98/stat98_6/stat98_6.htm`, 16-August-2017

[50] Lidong Wang, "Big data in intrusion Detection Systems and Intrusion Prevention Systems", Journal of Computer Networks, August-2017

[51] LINCOLN LABORATORY, DARPA INTRUSION DETECTION EVALUATION, `https://ll.mit.edu/ideval/data/1998data.html`, 23-August-2017

[52] Madhusudhanan Chandrasekaran, Krishnan Narayanan and Shambhu Upadhyaya, *Phishing E-mail Detection Based on Structural Properties*, "Department of Computer Science and Engineering"

[53] Mahmoud Sammour, Burairah Hussin, Mohd Fairuz Iskandar Othman, *Comparative Analysis for Detecting DNS Tunneling Using Machine Learning Techniques*, International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 22 (2017) pp. 12762-12766

[54] Martin Buhmann, Radial basis function, `http://www.scholarpedia.org/article/Radial_basis_function`, `2010`, 16-August-2017

[55] Manos Antonakakis, Roberto Perdisci, Yacin Nadji, Nikolaos Vasiloglou, Saeed Abu-Nimeh, Wenke Lee and David Dagon, "From Throw-Away Traffic to Bots : Detecting the Rise of DGA-Based Malware, " *Presented as part of the 21st USENIX Security Symposium (Usenix Security 12)*, pp. 491-506, 2012

[56] Michael Crawford, Taghi M. Khoshgoftaar, Joseph D. Prusa, Aaron N. Richter and Hamzah Al Najada, *Survey of review spam detection using machine learning techniques*, "Journal of Big Data 2015"

[57] Michael FOESSEL & Antoine GARAPON, "Biométrie : les nouvelles formes de l'identité", *Esprit*, July-August 2006

[58] Neethu B, "Adaptive Intrusion Detection Using Machine Learning", International Journal of Computer Science and Network Security, 3, March-2013

[59] Nutan Farah H., Musharaat Rafni, Abdur Rahman O., Faisal Muhammad S., Md Avished Khan H., Dewa Mhd F., "Application of Machine Learning approches in intrusion Detection System :A Survey", International Journal of Advanced Research in Artificial Intelligence, Vol 4, 2015

[60] OpenDNS, `https://www.phishtank.com/`

[61] Pavani Bharathula, N. Mridula Menon, *Equitable Machine Learning Algorithms to Probe Over P2P Botnets*

[62] Pawel Kobojek, Khalid Saeed, "Application of Recurrent Neural Networks for User Verification based on Keystroke Dynamics", *Faculty of Mathematics and Information Sciences*

[63] Philippe Beraud - MSFT, L'apprentissage automatique (Machine Learning), comment ça marche ?, `https://blogs.msdn.microsoft.com/big_data_france/2014/06/05/lapprentissage-automatique-machine-learning-comment-a-marche/`, 5-June 2014, 24-April 2017

[64] Pijush Barthakur, Manoj Daha, Mrinal Kanti Ghose, *An Efficient Machine Learning Based Classification Scheme for Detecting Distributed Command & Control Traffic of P2P Botnets*, November 2013

[65] Pranita Wankhede and Dharmpal Doye, "Support Vector Machines for Fingerprint Classification", Department of Electronics and Telecommunication, S.G.G.S. Institute of Engineering and technology

[66] Prabakar Muniyandi Amuthan, Rajeswari, R Rajaram, " Network Anomaly Detection by Cascading K-Means Clustering and C4.5 Decision Tree algorithm.", *Procedia Engineering*. 30. 174-182.

[67] Praveer Mansukhani, Sergey Tulyakov and Venu Govindaraju, "Using Support Vector Machines to Eliminate False Minutiae Matches during Fingerprint Verification", Center for Unified Biometrics and Sensors (CUBS), 23-August-2017

[68] Promise Molale, Bhekisipho Twala and Solly Seeletse, "FINGERPRINT PREDICTION USING STATISTICAL AND MACHINE LEARNING METHODS", *CIC Express Letters*, Volume 7, Number 2, February 2013

[69] Qu'est-ce que le machine learning ?, https://openclassrooms.com/courses/initiez-vous-au-machine-learning/qu-est-ce-que-le-machine-learning

[70] Quamar Niyaz, Weiqing Sun, Ahmad Y Javaid, and Mansoor Alam, "A Deep Learning Approach for Network Intrusion Detection System", College Of Engineering The University of Toledo

[71] R.Kohavi &Gorge H.John, "Wrappers for feature subset selection", *Artificial Intelligence 97*, pages 273-324, 1997

[72] Raj Kumar Nepali, Yazan Alshboul, Yong Wang, *Detecting malicious short URLs on Twitter*

[73] Reuben Smith, Nathalie JapKowciz, Maxwell Dondo, Peter Mason *Using Unsupervised Learning for Network alert Correlation*

[74] Ritu, Rishabh Kaushal, *Machine Learning Approach for Botnet Detection*

[75] Robin Genuer, Jean-Michel Poggi Arbres CART et Forets aléatoires Importance et sélection de variables

[76] Romil Rawat, Shailendra Kumar Shrivastav, *SQL injection attack Detection using SVM*, "International Journal of Computer Applications (0975 – 8887)", Volume 42– No.13, March 2012

[77] Renuka Joshi, Accuracy, Precision, Recall & F1 Score : Interpretation of Performance Measures, http://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/, SEPTEMBER 9- 2016, 30-January.2017

[78] Sandhya Peddabachigari, Ajith Abraham, Johnson Thomas, "Intrusion Detection Systems Using Decision Trees and Support Vector Machines" Department of Computer Science, Oklahoma State University, USA

[79] Saurav Kaushik, Introduction to Feature Selection methods with an example (or how to select the right variables ?), https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/, DECEMBER 1- 2016, 15-Aug.2017

[80] Saurav Kaushik, An Introduction to Clustering and different methods of clustering, https://www.analyticsvidhya.com/blog/2016/11/an-introduction-to-clustering-and-different-methods-of-clustering/, 3-November-2016, 23-August-2017

[81] Scikit-learn developers (BSD License), http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html, 2017, 16-August-2017

[82] Scikit-learn developers (BSD License), http://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html, 2017, 16-August-2017

[83] Scikit-learn developers (BSD License), http://scikit-learn.org/stable/modules/cross_validation.html, 2017, 16-August-2017

[84] Scikit-learn developers(BSD License), Naive Bayes, http://scikit-learn.org/stable/modules/naive_bayes.html, 2017, 23-August-2017

[85] SETIONO R., AND LUI H., " Neural-network feature selector", *IEEE Transactions on Neural Networks 8, 3* , pages 654-662, 1997

[86] Shalini Chaurasia, C. Rama Krishna, "Performance Analysis of Supervised Learning Based Intrusion Detection System", *Computer Science & Engineering Department NITTTR*

[87] SLDM III, Hastie and Tibshirani- February 25, 2009, "K-fold Cross validation", *Cross validation and boostrap* , 25-Feb-2009

[88] Srinivas Mukkamala, Guadalupe Janoski, Andrew Sung, "Intrusion Detection Using Neural Networks and Support Vector Machines" , Department of Computer Science

[89] Standard Reference Data NIST, NIST Special Database 4, https://www.nist.gov/srd/nist-special-database-4, 27-August-2010, 21-July-2017, 21-August-2013

[90] Statsoft Support Vector Machines (SVM) Introductory Overview, http://www.statsoft.com/textbook/support-vector-machines, 2017, -23-August-2017

[91] Stéphane Vialle, "Objectifs et principes du Machine Learning", Big Data : Informatique pour les données et calculs massifs, 2017

[92] Stephen Robertson *Understanding Inverse Document Frequency : On theoretical arguments for IDF Microsoft Research, 7 JJ Thomson Avenue Cambridge CB3 0FB UK (and City University, London, UK)*

[93] Supervised Learning Workflow and Algorithms , https://nl.mathworks.com/help/stats/supervised-learning-machine-learning-workflow-and-algorithms.html, 16-August-2017

[94] T.Subbulakshmi, George Mathew, Dr S. Mercy Shalinie "Real Time classification and clustering of IDS Alert using Machine learning algortihms", International Journal of Artificial Intelligence & Applications (IJAIA), Vol. 1, No.1, January 2010

[95] Thiago S.Guzella, Walmir M.Caminhas, *A review of machine learning approaches to Spam filtering*, Volume 36, Issue 7, September 2009, Pages 10206-10222

[96] Tianyu Wang, Li-Chiou Chen, *Detecting Algorithmically Generated Domains Using Data Visualization and N-Grams Methods*, Proceedings of Student-Faculty Research Day, CSIS, Pace University, 5th-May- 2017

[97] Tom M Michell, *Machine Learning*, MIT Press and The McGraw-Hill Companies-inc., 1997

[98] Vaibhav Nivargi Mayukh Bhaowal Teddy Lee, "Machine Learning Based Botnet Detection", *CS 229 Final Project Report*

[99] Vinnie Monaco, Dataset, `http://www.vmonaco.com/datasets`, 23-August-217

[100] William Cukierski, `https://www.kaggle.com/wcukierski/enron-email-dataset`, 2016, 16-August-2017

[101] Vipin Das, Vijaya Pathak, Sattvik Sharma, Sreevathsan, MVVNS.Srikanth, Gireesh Kumar T, "NETWORK INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING ALGORITHMS", International Journal of Computer Science & Information Technology (IJCSIT), Vol 2, 6, December 2010

[102] Willy CHARON, INTRODUCTION AUX RÉSEAUX DE NEURONES : LE NEURONE FORMEL, `http://systemes-meca-actifs.pagesperso-orange.fr/5-RN/2-IntroRN/IntroRN.xhtml`, 17-August-2017

[103] Yannis Chaouche, Décomposez l'apprentissage d'une régression linéaire, `https://openclassrooms.com/courses/initiez-vous-au-machine-learning/tp-decomposez-l-apprentissage-d-une-regression-lineaire`, 2-August-2017, 16-August-2017

[104] Ying Zhao, "Learning User Keystroke Patterns for Authentication", *World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol :2, No :2*, 2008

[105] Younes Bennan, "Apprentissage par réseau de neurones artificiels ", *Artificial Network Learning*, 2014

[106] Yuma Sato, Yoshitaka Nakamura, Hiroshi Inamura and Osamu Takahashi, *A Proposal of Malicious URLs Detection based on Features Generated by Exploit Kits*, International Workshop on Informatics (IWIN 2016)

[107] Z.ISMAIL, A.JANTAN, "A Review of Machine Learning Application in Botnet Detection System", *SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)*, 17-October-2016