



UNIVERSITE SAAD DAHLEB 1 DE BLIDA

FACULTE DES SCIENCES

DEPARTEMENT INFORMATIQUE



*Domaine : LMD Mathématiques informatique*

*Spécialité : Master Académique*

*Option: Système Informatique Et Réseau*

## MEMOIRE DE MASTER

---

# COCEPTION ET MISE EN ŒUVRE D'UN SYSTEME DE CLASSIFICATION DU TRAFIC RESEAU CRYPTÉ

---

Par :

**MINA Madina**

**SOUKEHAL Roumaissa**

**Devant le jury composé de:**

Mr. OULD KHAOUA Mohammed.

Mr. DOUGA Yacine.

Mde. BOUTOUMI Bachira.

Mr. BOUDHAB Tarek.

Président

Examineur

Promoteur

Encadreur

2019/2020

# Résumé

A l'instar des autres entreprises du secteur économique, Sonatrach, la plus importante entreprise du pays qui opère dans le secteur de l'énergie, souhaite gérer et mieux sécuriser ces importantes infrastructures informatiques et ses systèmes d'informations. Un éventuel dysfonctionnement de ces derniers peut porter de graves préjudices économiques à l'entreprise et au pays.

La classification du trafic réseau au sein de l'entreprise joue un rôle important. Elle permet de catégoriser le trafic, faire des statistiques, appliquer une politique de qualité de service adéquate, détecter les intrusions, etc. Mais vu la vulgarisation des techniques de chiffrement, la classification du trafic réseau chiffré par les approches traditionnelles est devenue ardue. De ce fait, l'administration du trafic réseau est devenue un problème.

Pour pallier à ce problème, qui est devenu une priorité pour les administrateurs réseaux à Sonatrach, il fallait trouver de nouvelles approches efficaces et rapides pouvant catégoriser des paquets chiffrés. A cet effet, nous avons proposé des solutions basées sur des techniques statistiques. Elles utilisent des algorithmes d'apprentissage automatique (Machine learning), apprentissage profond (Deep learning) et des données pour générer des modèles. Le meilleur modèle sera utilisé pour classer le trafic. Pour l'implémentation de notre solution, nous avons utilisé différents algorithmes d'apprentissage, tel que ResNet, Random forest, Naive Bayes, etc. Pour améliorer nos résultats nous avons utilisé différentes techniques de sélection d'attributs (filter, wrapper).

Après benchmarking profond la meilleure performance atteinte est 99% en utilisant Random Forest et kneighborsclassifier.

**Mots clés :** Classification du trafic, trafic réseau chiffré, Apprentissage profond, Apprentissage Automatique, sélection des attributs.

# Abstract

Like other companies in the economic sector, Sonatrach, the country's largest company operating in the energy sector, wants to manage and better secure its important IT infrastructure and information systems. A possible malfunctioning of the latter can cause serious economic damage to the company and the country.

The classification of network traffic within the company plays an important role. It allows to categorize the traffic, to make statistics, to apply an adequate quality of service policy, to detect intrusions, etc. But with the popularization of encryption techniques, classifying encrypted network traffic using traditional approaches has become difficult. As a result, the administration of network traffic has become a problem.

To overcome this problem, which has become a priority for network administrators at Sonatrach, it was necessary to find new efficient and fast approaches that could categorize encrypted packets. To this end, we proposed solutions based on Data statistical techniques. They use machine learning, deep learning and data to generate models. The best model will be used to classify the traffic. For the implementation of our solution, we used different learning algorithms, such as ResNet, Random forest, Naive Bayes, ect. To improve our results we have used different attribute selection techniques (filter, wrapper).

After deep benchmarking the best performance achieved is 99% using Random Forest and kneighborsclassifier.

**Keywords:** Traffic classification, Encrypted network traffic, Deep learning, Automatic learning, Feature selection.

## ملخص

مثل الشركات الأخرى في القطاع الاقتصادي ، ترغب سوناطراك ، أكبر شركة في البلاد تعمل في قطاع الطاقة ، في إدارة وتأمين البنى التحتية لتكنولوجيا المعلومات وأنظمة المعلومات بشكل أفضل. إذا فشلوا في العمل ، فقد يتسببون في أضرار اقتصادية خطيرة للشركة وللبلد يلعب تصنيف حركة مرور الشبكة داخل المؤسسة دورًا مهمًا. يتم استخدامه لتصنيف حركة المرور ، وتجميع الإحصاءات ، وتطبيق سياسة جودة الخدمة المناسبة ، واكتشاف التدخلات ، وما إلى ذلك. ولكن مع تعميم تقنيات التشفير ، أصبح تصنيف حركة مرور الشبكة المشفرة بواسطة الأساليب التقليدية أمرًا صعبًا. نتيجة لذلك، أصبحت إدارة حركة مرور الشبكة مشكل

للتغلب على هذه المشكلة ، التي أصبحت أولوية لمسؤولي الشبكات ، كان لابد من إيجاد طرق جديدة فعالة وسريعة Sonatrach في شركة يمكنها تصنيف الحزم المشفرة. تحقيقًا لهذه الغاية ، اقترحنا حلاً تستند إلى تقنيات علوم البيانات. يستخدمون التعلم الآلي والتعلم العميق وخوارزميات البيانات لإنشاء النماذج. سيتم استخدام أفضل نموذج لترتيب حركة المرور. لتنفيذ الحل الخاص بنا ، استخدمنا Naive و Random Forest و ResNet خوارزميات تعليمية مختلفة ، مثل ، إلخ. لتحسين نتائجنا ، استخدمنا تقنيات مختلفة لاختيار Bayes (السمات) مرشح ، غلاف

**الكلمات الرئيسية:** تصنيف حركة المرور، حركة مرور الشبكة المشفرة، التعلم العميق، التعلم التلقائي، اختيار الميزة

## **Remerciements**

*Nous remercions Dieu le tout puissant de nous avoir  
Donné la santé et la volonté d'entamer et de terminer ce  
Mémoire.*

*Ce travail ne serait pas aussi riche et n'aurait pas pu  
Voir le jour sans l'aide et l'encadrement du Docteur  
Boudhab tarek, nous la remercions chaleureusement  
Pour la qualité de son encadrement exceptionnel, pour  
Sa patience, sa rigueur et sa disponibilité durant toute  
Notre préparation de ce mémoire.*

*Nos sincères remerciements s'adressent également à  
Nos promotrices Docteur BOUTOUMI Bachira et Docteur  
ZAHRA fatma zohra pour son aide pratique, son soutien  
moral et ses  
Encouragements.*

*Nous sommes conscients de l'honneur que nous a fait  
Les membres de jury d'avoir accepté d'examiner ce  
Travail, nous les remercions énormément.*

*Un grand merci à Madame DIF Nassima et à  
Toute l'équipe de Sonatrach.*

*Nos profonds remerciements vont également à  
Toutes les personnes qui nous ont aidé et soutenu de près  
Ou de loin.*

## **Dédicaces**

*Je dédie ce modeste travail À*

*MES TRÈS CHÈRS PARENTS*

*Pour leur amour inestimable, leurs soutiens et leurs  
Sacrifices. Rien au monde ne vaut les efforts fournis jour et  
Nuit pour mon éducation et ma bien être, que Dieu vous  
Garde en bonne santé et vous prête longue vie.*

*Ma sœur Soumia et mon frère.*

*Toute ma famille.*

*Mon encadreur Mr BOUDHAB TAREK.*

*Mon Binôme madina et sa sœur marwa et à toute la  
Famille MINA.*

*Mes meilleures amies.*

*Je ne peux nommer ici toutes les personnes qui de près ou de  
Loin m'ont aidés et encouragés mais je les en remercie  
vivement.*

***ROUMAISSA.***

## **Dédicaces**

*A ma tendre Mère : Tu représentes pour moi la  
Source de tendresse et l'exemple de dévouement qui n'a pas  
Cessé de m'encourager. Tu as fait plus qu'une mère puisse  
Faire pour que ses enfants suivent le bon chemin dans leur  
vie et leurs études.*

*A mon très cher Père Toufik : Aucun dédicace  
Ne serait exprimé l'amour, l'estime et le respect que j'ai  
Toujours pour toi. Rien au monde ne vaut les efforts  
Fournis jour et nuit pour mon éducation et mon bien être.  
Ce travail est le fruit de tes sacrifices que tu as consenti  
Pour mon éducation et ma formation le long de ces années.*

*A mes très chers Frères, sœurs et beaux-frères.*

*A mes copines et toute ma famille.*

*A ma sœur et mon binôme Roumaïssa et à toute la  
famille SOUKEHAL.*

*Et à tous ceux qui ont contribué de près ou de loin  
Pour que ce projet soit réalisé, je vous dis merci.*

**MADINA**

# Table des Matières

<i>Liste des Tableaux</i> .....	8
<i>Liste des Figures</i> .....	9
<i>Liste des Abréviations</i> .....	11
INTRODUCTION.....	1
1 Introduction et problématique .....	1
2 Objectif de travail.....	2
3 Organisation du mémoire.....	3
PARTIE I: LES PRINCIPES FONDAMENTAUX.....	4
DE LA CLASSIFICATION DU TRAFIC.....	4
RÉSEAU .....	4
Chapitre 1 : Machine Learning .....	5
Et Deep Learning .....	5
1 Introduction .....	5
2 Machine Learning .....	6
2.1 L'apprentissage supervisé .....	6
2.1.1 Régression .....	7
2.1.2 Classification.....	8
2.2 L'apprentissage non supervisé .....	12
3 Transfer Learning .....	12
3.1 Les méthodes de transfer learning.....	13
3.2 Apprentissage par transfert avec les données d'images.....	13
4 Les réseaux de neurones artificiels et deep Learning .....	14
4.1 Les réseaux de neurones artificiels.....	14
4.2 Deep Learning .....	15
5 Les algorithmes de Deep Learning .....	16
5.1 Les réseaux de neurones convolutifs.....	16
5.1 Réseau neuronal résiduel (ResNet) .....	20
6 Cycle de vie de l'implémentation des algorithmes .....	21
6.1 Prétraitement .....	21
6.2 La modélisation .....	25
6.3 La phase d'apprentissage .....	26
6.4 Validation.....	26
6.5 Performance du modèle.....	27
7 Conclusion.....	28
Chapitre 2: Classification.....	29
Du Trafic Réseau Chiffré.....	29
1 Introduction .....	29



2	Trafic réseau.....	29
2.1	Paquet réseau.....	30
3	Flux de donnée.....	31
4	Classification du trafic réseau.....	31
4.1	Classification basée sur les ports.....	32
4.2	Classification par l'inspection de charge.....	32
4.3	Approche comportementale .....	33
4.4	Approche statistique.....	34
5	Définition d'un VPN .....	34
6	Les travaux connexes .....	35
7	Conclusion.....	39
	PARTIE II : EXPERIMENTATON .....	40
	Chapitre 3 : Méthodologie .....	41
1	Introduction .....	41
2	DataSet.....	41
3	Outil de la réalisation.....	43
4	Description du système.....	44
5	Preprocessing.....	45
6	La sélection d'attributs .....	46
7	Cross-validation .....	46
8	Les algorithmes de classification.....	47
9	Métriques utilisés.....	48
10	Conclusion.....	49
	Chapitre 4: Résultats .....	50
	Expérimentaux et discussion.....	50
1	Introduction .....	50
2	Première expérience .....	50
3	Deuxième expérience.....	51
4	Troisième expérience.....	53
5	Quatrième expérience.....	56
6	Cinquième expérience.....	59
7	Conclusion.....	62
	CONCLUSION .....	63
1	Contributions et résumé des résultats expérimentaux .....	63
2	Limites et travaux futurs.....	64
	REFERENCES .....	65

# Liste des Tableaux

<b>Tableau 2.1 :</b> Travaux liés à la classification machine learning.....	37
<b>Tableau 2.2 :</b> Travaux liés à la classification deep learning.....	38
<b>Tableau 3.1 :</b> Contenu de l'ensemble de données de trafic réseau UNC ISCX [74].....	41
<b>Tableau 3.2 :</b> Nombre d'occurrence de classes.....	42
<b>Tableau 3.3 :</b> Configuration de l'apprentissage automatique.....	47
<b>Tableau 4.1 :</b> Résultats de classification machine learning sans sélections d'attributs avec Hold out.....	50
<b>Tableau 4.2 :</b> Résultats de classification machine learning sans sélections d'attributs avec 10_fold cross validation. ....	52
<b>Tableau 4.3 :</b> Résultats de classification machine learning en utilisant ma méthode filter de sélections d'attributs avec Hold out.....	54
<b>Table 4.4 :</b> Résultats de classification machine learning en utilisant ma méthode wrapper de sélections d'attributs avec Hold out. ....	55
<b>Tableau 4.5 :</b> Résultats de classification machine learning en utilisant ma méthode filter de sélections d'attributs avec K_fold cross validation.....	57
<b>Tableau 4.6 :</b> Résultats de classification machine learning en utilisant ma méthode wrapper de sélections d'attributs avec 10_fold cross validation.....	58
<b>Tableau 4.7:</b> Résultats de classification transfer learning avec Hold out.....	60

# Liste des Figures

<b>Figure 1.1</b> : Les types d'apprentissage automatique.....	6
<b>Figure 1.2</b> : Régression logistique.....	8
<b>Figure 1.3</b> : Limites de décisions multiples .....	8
<b>Figure 1.4</b> : Frontière de décision avec les vecteurs desoutien .....	9
<b>Figure 1.5</b> : Régression de l'arbre de décision.....	10
<b>Figure 1.6</b> : Le classificateur Random forest.....	10
<b>Figure 1.7</b> : Exemple de classification KNN.....	12
<b>Figure 1.8</b> : Types de Transfer Learning.....	13
<b>Figure 1.9</b> : Un réseau de neurone artificiel.....	14
<b>Figure 1.10</b> : Deep Learning. ....	15
<b>Figure 1.11</b> : Des réseaux simples aux réseaux profonds.....	15
<b>Figure 1.12</b> : Comparaison entre l'apprentissage automatique (a) et Deep Learning (b).....	16
<b>Figure 1.13</b> : Architecture de réseau neuronal convolutif .....	16
<b>Figure 1.14</b> : Convolution dans CNN.....	17
<b>Figure 1.15</b> : Pooling.....	19
<b>Figure 1.16</b> : La formule de ResNet. ....	21
<b>Figure 1.17</b> : One hot encoding. ....	22
<b>Figure 1.18</b> : Intger encoding. ....	23
<b>Figure 1.19</b> : Exemple de valeurs aberrantes.. ....	24
<b>Figure 1.20</b> : Un exemple de conversion de données tabulaires en images avec two-dimensional Embedding .....	25
<b>Figure 2.1</b> : Réseaux de données. ....	30
<b>Figure 2.2</b> : Les Approcheche de la classification.....	31
<b>Figure 2.3</b> : VPN.....	34
<b>Figure 2.4</b> : Une Communication VPN.....	35
<b>Figure 2.4</b> : Les méthodes de cassification.....	36
<b>Figure 3.1</b> : Capture d'écran de la plateforme en ligne Google colab. ....	44
<b>Figure 3.2</b> : Mécanisme du système.....	45
<b>Figure 3.3</b> : Le taux d'importance des attributs sélectionnées.....	46
<b>Figure 3.4</b> : Configuration 10-fold cross validation.. ....	46
<b>Figure 3.5</b> : Configuration Hold out.....	46
<b>Figure 3.6</b> : Configuration d'une classification repport.....	48
<b>Figure 3.7</b> : Exemple d'une matrice de confusion.....	49
<b>Figure 4.1</b> : Comparaison entre f_mesures des algorithmes machine learning avec une répartition	

hold out.....	51
<b>Figure 4.2</b> : Comparaison entre f_mesures des algorithmes machine learning avec une répartition 10_fold.....	53
<b>Figure 4.3</b> : Comparaison entre f_mesures des algorithmes machine learning en utilisant hold out et la méthode filter.....	55
<b>Figure 4.4</b> : Comparaison entre f_mesures des algorithmes machine learning en utilisant hold out et la méthode wrapper.....	56
<b>Figure 4.5</b> : Comparaison entre f_mesures des algorithmes machine learning en utilisant 10_fold et la méthode filter.....	58
<b>Figure 4.6</b> : Comparaison entre f_mesures des algorithmes machine learning en utilisant 10_fold et la méthode wrapper.....	59
<b>Figure 4.7</b> : Error Rate.....	60
<b>Figure 4.8</b> : La comparaison entre tous les algorithmes utilisés. ....	61

## *Liste des Abréviations*

<b>QoS</b>	: Quality of service.
<b>OSI</b>	: Open Systems Interconnection.
<b>DPI</b>	: L'inspection des paquets de données.
<b>TLS</b>	: Transport Layer Security.
<b>ISCX</b>	: Installation Support Center of Expertise.
<b>VPN</b>	: Virtual Private Network.
<b>CSV</b>	: Comma-Separated Values.
<b>TCP/IP</b>	: Transmission Control Protocol / Internet Protocol.
<b>ICMP</b>	: Internet Control Message Protocol.
<b>P2P</b>	: Peer to peer.
<b>SPI</b>	: Stochastic Packet Inspection.
<b>CNN</b>	: Convolutional Neural Network.
<b>LSTM</b>	: Long short-term memory.
<b>NTC</b>	: Negative Temperature Coefficient.
<b>SVM</b>	: Support Vector Machine.
<b>KNN</b>	: k-nearest neighbor.
<b>IA</b>	: Intelligence Artificielle.
<b>ML</b>	: machine learning.
<b>1D CNN</b>	: One dimensionnel neural network.
<b>NLP</b>	: Neuro-Linguistic Programming.
<b>ResNet</b>	: Residual Network.
<b>TP</b>	: True Positive.
<b>TN</b>	: True Negative.
<b>FP</b>	: False Positive.
<b>FN</b>	: False Negative.
<b>UNB</b>	: Université de new Brunswick.
<b>RF</b>	: Randon Forste.
<b>LR</b>	: LogisticRegression.
<b>NB</b>	: Naive Bays.
<b>IANA</b>	: Internet Assigned Numbers Authority.

# INTRODUCTION

## 1 Introduction et problématique

La classification du trafic réseau est une tâche importante dans les réseaux de communication modernes [1]. En raison de la croissance rapide des demandes de trafic à haut débit, pour gérer correctement les ressources du réseau, il est essentiel de reconnaître les différents types d'applications utilisant les ressources du réseau. Par conséquent, la classification précise du trafic est devenue l'une des conditions préalables aux tâches avancées de gestion de réseau telles que la fourniture d'une qualité de service (QoS) appropriée, la détection des anomalies, la tarification, etc. La classification du trafic a suscité beaucoup d'intérêt dans les milieux universitaires et les activités industrielles liées aux réseaux.

Selon la différence de couche OSI, les techniques de chiffrement du trafic peuvent être divisées en chiffrement de la couche application, chiffrement de la couche présentation et chiffrement de la couche réseau [2]. Le chiffrement de la couche application signifie que les applications mettent en œuvre leurs propres protocoles pour la transmission sécurisée des données dans la couche application (par exemple BitTorrent ou Skype), et il est également appelé chiffrement ordinaire dans certains documents. Le chiffrement de la couche présentation et le chiffrement de la couche réseau signifient que les applications cryptent l'ensemble des paquets de la couche supérieure, et les techniques typiques sont TLS et IPsec. Certaines technologies de tunnel comme le VPN sont basées sur ces techniques. Ce type de chiffrement est également appelé encapsulation de protocole. Dans certains cas, le trafic chiffré par le chiffrement normal peut être chiffré davantage par l'encapsulation de protocole (par exemple, le trafic Skype par VPN).

L'émergence de nouvelles applications ainsi que les interactions entre les différents composants de l'Internet ont considérablement accru la complexité et la diversité de ce réseau, ce qui rend la classification du trafic problème difficile en soi.

Les techniques de classification du trafic ont beaucoup évolué au fil du temps. La première approche, la plus simple, consiste à utiliser les numéros de port. Toutefois, sa précision a diminué car les applications les plus récentes utilisent soit des numéros de port bien connus pour dissimuler leur trafic, soit n'utilisent pas de numéros de port standard

enregistrés. La prochaine génération de classification de trafic, basée sur la charge utile ou l'inspection des paquets de données (DPI), se concentre sur la recherche de modèles ou mots clés dans les paquets de données. Ces méthodes sont uniquement applicables au trafic non chiffré et elle a des frais de calcul élevés. Comme Il en résulte une nouvelle génération de méthodes, basées sur les statistiques de flux. Ces méthodes reposent sur des séries statistiques ou temporelles qui leur permettent de traiter des données chiffrées et non chiffré. [15]

Sur la base de la taxonomie de classification du trafic réseau, il s'agit de deux approches d'apprentissage automatique, dont le déroulement général est le suivant : premièrement, conception manuelle des caractéristiques du trafic (par exemple, caractéristiques de flux ou de paquets), deuxièmement, extraction et sélection de ces caractéristiques sous forme de trafic brut, afin de classifier le trafic avec ces caractéristiques par un classificateur conçu manuellement (arbre de décision, Naive Bayes, ...ect).

L'apprentissage approfondi permet d'éviter la sélection de fonctionnalités par un expert du domaine car il sélectionne automatiquement les fonctionnalités par le biais de la formation. Cette caractéristique fait de l'apprentissage approfondi une approche hautement souhaitable pour la classification du trafic, en particulier lorsque de nouvelles classes apparaissent constamment et que les modèles des anciennes classes évoluent. Une autre caractéristique importante de l'apprentissage approfondi est qu'il a une capacité d'apprentissage considérablement plus élevée, et peut donc apprendre des modèles très compliqués. [15]

## 2 Objectif de travail

Ce mémoire porte sur la construction d'un système de classification capable d'indiquer si un trafic est un VPN ou NON-VPN utilisant un ensemble de données de trafic réseau. Ce problème a été abordé par la communauté des chercheurs à l'aide de diverses méthodologies. La plupart d'entre elles se sont concentrées sur la recherche des caractéristiques pertinentes et d'un classificateur approprié pour améliorer les performances de systèmes de classification. Nous avons effectué des comparaisons expérimentales approfondies entre les modèles de classification des trafics réseaux basées sur l'ensemble de données ISCX VPN NON-VPN.

Afin d'atteindre notre objectif de conception et implémentation d'un modèle de classification du trafic réseau VPN ou non VPN en fonction de type d'application, nous avons Prétraités notre DataSet pour qu'on puisse les classifier par plusieurs algorithmes d'apprentissage automatique et d'apprentissage profond afin de trouver un modèle qui donne

des meilleurs performances.

### **3 Organisation du mémoire**

Ce mémoire se compose de deux parties principales. La première partie couvre les notions de pointe nécessaires à la compréhension des idées développées dans ce mémoire. Nous présentons dans le chapitre 1 certains concepts de deep Learning et machine Learning, en fournissant une brève description des classificateurs supervisés, des mesures d'évaluation et des tests statistiques invoqués dans ce travail. Dans le chapitre 2, nous passons en revue un aperçu sur la classification du trafic réseau chiffré. La seconde moitié de ce mémoire décrit la méthodologie que nous avons suivie pour comparer les systèmes de classifications de trafic réseau chiffré. Nous fournissons au chapitre 3 une description détaillée du dispositif expérimental, y compris le cadrage, le fenêtrage et le paramétrage. Dans le chapitre 4, nous présentons les résultats obtenus à l'aide de tableaux de performances et de graphiques basés sur des statistiques. Plus important encore, nous avons étayé notre discussion en nous basant sur des tests statistiques bien connus. Enfin, nous concluons en résumant les contributions de ce mémoire, les limites et les travaux futurs.



# **PARTIE I: LES PRINCIPES FONDAMENTAUX DE LA CLASSIFICATION DU TRAFIC RÉSEAU**

Dans cette partie, nous expliquons les notions nécessaires à la compréhension des idées développées dans ce mémoire. Elle est composée de deux chapitres. Dans le chapitre 1, nous présentons en revue certains concepts de deep Learning et machine Learning, en fournissant une description des classificateurs supervisés, et la relation entre les modèles de deep Learning et machine Learning avec la classification du trafic réseau. Nous donnons au chapitre 2 un aperçu sur la classification du trafic réseau. Plus précisément, les méthodes de la classification du trafic réseau ainsi que les caractéristiques d'un flux de données. En outre, nous présentons les différents travaux qu'ils ont fait.

# Chapitre 1 : Machine Learning Et Deep Learning

## 1 Introduction

Dans le domaine de l'intelligence artificielle, la machine Learning est la technologie qui permet aux machines d'apprendre seules à partir de données fournies. Ces algorithmes permettent aujourd'hui de résoudre des équations ou des cas qui semblaient être insolubles. [3].

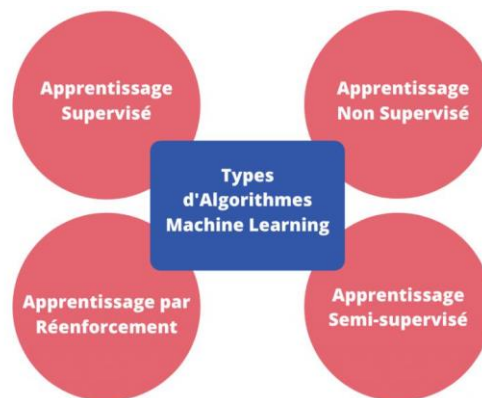
En revanche, le Deep Learning est un sous-ensemble de la machine Learning. Il se compose d'un ensemble de modèles appelés réseaux de neurones, qui contrairement aux modèles d'apprentissage automatique, ce qui signifie qu'ils ont besoin d'une grande quantité de données pour bien fonctionner. Il existe de nombreuses approches d'apprentissage automatique qui se répartissent en trois grandes catégories d'apprentissage supervisé, non supervisé et semi supervisé. Nous nous concentrons largement sur l'approche d'apprentissage supervisé car elle est la plus fréquemment utilisée pour la classification du trafic réseau. Il consiste à construire des modèles pour apprendre une correspondance entre les entités extraites et les étiquettes de classe pour les classes des trafics où les étiquettes sont prédéfinies à l'avance et déterminées à partir des annotations de référence.

Le reste de ce chapitre est structuré comme suit. Dans la section 2, nous présentons C'est quoi la machine Learning. Dans la section 3, on passe en revue le principe de transfer learning. De plus, dans la section 4, nous expliquons le principe de réseaux de neurones et le Deep Learning, puis nous discutons dans la section 5 les algorithmes d'apprentissage. Ensuite dans la section 6, nous présentons le cycle de vie d'une implémentation de machine Learning. Enfin, à la section 7, nous concluons ce chapitre en résumant les principaux concepts que nous avons définis.

## 2 Machine Learning

L'apprentissage automatique fait référence au développement, à l'analyse et à l'implémentation des méthodes qui permettent à une machine d'évaluer grâce à un processus d'apprentissage, ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyenne algorithmique plus classique. L'objectif de l'apprentissage automatique est d'extraire et exploiter automatiquement l'information présente dans un jeu de données. [4]

Il existe plusieurs façons de faire apprendre à une machine (**Figure 1.1**). On distingue notamment l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement.



**Figure 1.1:** les types d'apprentissage automatique.

### 2.1 L'apprentissage supervisé

Dans le cas de l'apprentissage supervisé, le système est guidé dans son apprentissage. On lui indique le type de résultat à atteindre en le nourrissant d'exemples. Pour cela, on lui fournit des données d'entrée pour lesquelles le résultat est connu et communiqué au système. Le but est qu'il puisse ensuite généraliser ce qu'il a appris pour des données non connues. Par exemple, si le système doit apprendre à reconnaître des feuilles de vigne dans une image, on lui fournit des images où la feuille est signalée et où le label « feuille de vigne » est associé. On parle ainsi de données étiquetées ou annotées. Le jeu de données d'entraînement annotées permet au système de calculer ses erreurs en comparant ses résultats avec les résultats connus

et ainsi d'ajuster le modèle pour progresser. Une partie des données annotées (non utilisées pendant l'entraînement) pourra également servir à vérifier l'efficacité du modèle, une fois l'apprentissage terminé. [5]

En fonction du type de tâches, nous pouvons classer les modèles d'apprentissage automatique supervisé dans les types suivants [6]:

### 2.1.1 Régression

Dans la machine, l'apprentissage de la régression est un ensemble de problèmes où la variable de sortie peut prendre des valeurs continues.

#### 2.1.1.1 Régression logistique

La régression logistique en python s'agit d'une technique d'analyse d'un ensemble de données qui comporte une variable dépendante et une ou plusieurs variables indépendantes pour prédire le résultat dans une variable binaire, ce qui signifie qu'il n'y aura que deux résultats. La variable dépendante est de nature catégorique. La variable dépendante est également appelée variable cible et les variables indépendantes sont appelées prédicteurs. La régression logistique est un cas particulier de régression linéaire où nous ne prédisons le résultat que dans une variable catégorielle. Elle prédit la probabilité de l'événement à l'aide de la fonction logarithmique. Nous utilisons la fonction/courbe Sigmoid pour prédire la valeur catégorique. La valeur seuil détermine le résultat (gagnant/perdant). [7]

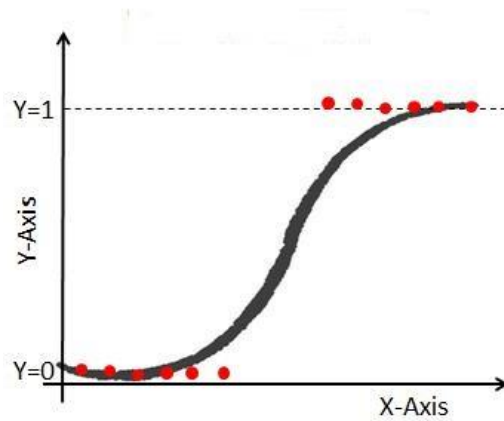
Équation de régression linéaire :

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n \quad (1)$$

- ✚  $y$  représente la variable dépendante qui doit être prédite.
- ✚  $\beta_0$  est l'intersection Y, qui est en fait le point sur la ligne qui touche l'axe des y.
- ✚  $\beta_1$  est la pente de la droite (la pente peut être négative ou positive selon la relation entre la variable dépendante et la variable indépendante).
- ✚  $X$  représente ici la variable indépendante qui est utilisée pour prédire notre valeur dépendante résultante.

– Fonction sigmoïde : 
$$p = \frac{1}{1+e^{-y}} \quad (2)$$

On Applique la fonction sigmoïde sur l'équation de régression linéaire (**Figure 1.2**).



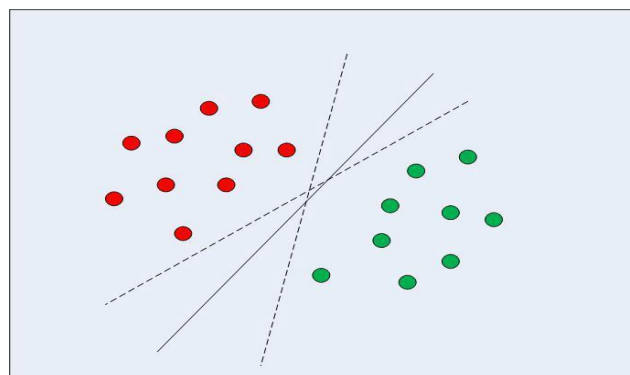
**Figure 1.2 :** Régression logistique. [7]

## 2.1.2 Classification

La classification est la tâche de prédire le type ou la classe d'un objet dans un nombre fini d'options. La variable de sortie pour la classification est toujours une variable catégorielle.

### 2.1.2.1 Machine à vecteur de support

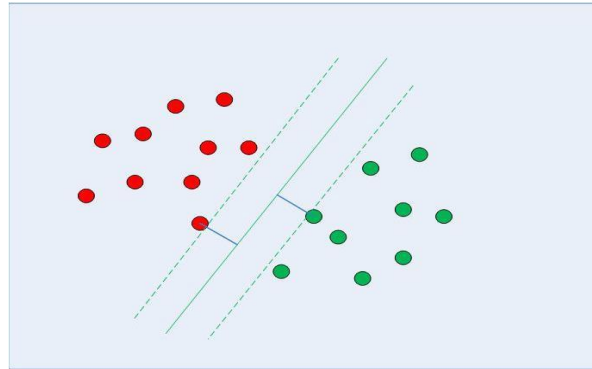
Dans le cas de données séparables linéairement en deux dimensions, comme le montre la figure, un algorithme d'apprentissage machine typique tente de trouver une limite qui divise les données de manière à minimiser l'erreur de classification. Si vous regardez attentivement la **figure 1.3**, il peut y avoir plusieurs limites qui divisent correctement les points de données. Les deux lignes pointillées ainsi qu'une ligne continue classent correctement les données.



**Figure 1.3 :** Limites de décisions multiples. [8]

Le SVM diffère des autres algorithmes de classification dans la mesure où il choisit la limite de décision qui maximise la distance par rapport aux points de données les plus proches de toutes les classes. Un SVM ne se contente pas de trouver une limite de décision ; il trouve la limite de décision la plus optimale.

La frontière de décision la plus optimale est celle qui a la marge maximale par rapport aux points les plus proches de toutes les classes. Les points les plus proches de la limite de décision qui maximisent la distance entre la limite de décision et les points sont appelés vecteurs de soutien, comme le montre la **figure 1.4**. La limite de décision dans le cas des machines à vecteurs supports est appelée classificateur de marge maximale, ou hyperplan de marge maximale. [8]



**Figure 1.4** : Frontière de décision avec les vecteurs de soutien. [8]

Il existe différents types des noyaux dans le SVM :

$$\text{+} \text{ Noyau linéaire : } K(X, Y) = X^T Y \quad (3)$$

$$\text{+} \text{ Noyau polynomial : } K(X, Y) = (\gamma \cdot X^T Y + r)^d, \gamma > 0 \quad (4)$$

$$\text{+} \text{ Noyau gaussien : } K(X, Y) = \exp(-\gamma \cdot \|X - Y\|^2), \gamma > 0 \quad (5)$$

$$\text{+} \text{ Noyau sigmoïde : } K(X, Y) = \tanh(\gamma \cdot X^T Y + r), \gamma > 0 \quad (6)$$

D'où  $r$ ,  $d$ , et  $\gamma$  sont des paramètres de noyau.

### 2.1.2.2 Arbre de décisions

Les arbres de décision (DT) sont une méthode d'apprentissage supervisé non paramétrique utilisée pour la classification et la régression. 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 des données. Par exemple, dans la **figure 1.5** ci-dessous, les arbres de décision apprennent à partir des données à se rapprocher d'une courbe sinusoïdale avec un ensemble de règles de décision "if-then-else". Plus l'arbre est profond, plus les règles de

décision sont complexes et plus le modèle est adapté. [9]

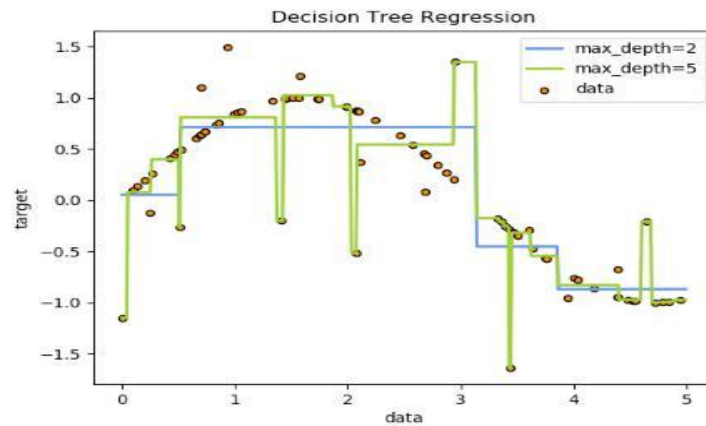


Figure 1.5 : régression de l'arbre de décision. [9]

### 2.1.2.3 Forêts aléatoires

Random Forest (algorithme des forêts d'arbres décisionnels) a été formellement proposé en 2001 par Leo Breiman et Adèle Cutler. Il aït partie des techniques d'apprentissage automatique. Cet algorithme combine les concepts de sous-espaces aléatoires et de «bagging». Le Random Forest effectue un apprentissage sur de multiples arbres de décision entraînés sur des sous-ensembles de données légèrement différents (Figure 1.6). [10]

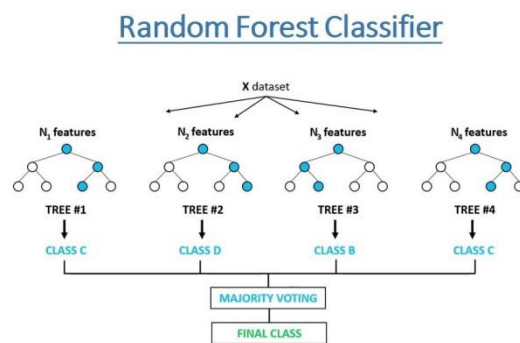


Figure 1.6 : Le classificateur Random forest.[10]

### 2.1.2.4 Naïve bayes

Pour une problématique de classification, la structure de naïve bayes a prouvé expérimentalement qu'elle était capable de donner de bons résultats [11]. L'hypomémoire de base de ce modèle est de supposer que toutes les observations étaient indépendantes les unes des autres conditionnellement à la variable classe, ce qui revient à une simplification de la loi jointe suivante :

$$P(A|B) = \frac{P(A).P(B|A)}{P(B)} \quad (7)$$

### 2.1.2.5 K-Nearest Neighbor

La règle de classification de K-plus proches voisins (KNN) est une méthode de classification puissante permettant la classification d'une instance inconnue en utilisant un ensemble d'instances de formation classées.

L'algorithme des k-plus proches voisins (k-Nearest Neighbours) permet la classification d'instances à partir des instances de formation les plus proches dans l'espace caractéristique.

Il s'agit d'un type d'apprentissage basé sur les instances, appelé aussi memory-based ou lazy-learning, car il n'y a pas d'apprentissage réel, les exemples d'apprentissage sont juste stockés en mémoire et réutilisés lors de la classification.

L'idée principale de l'algorithme est de prédire pour chaque nouvelle observation les k observations lui étant les plus similaires dans l'ensemble de données d'apprentissage. [12]

Lorsqu'on parle de voisin cela implique la notion de similarité ou de distance. La distance la plus souvent utilisé est la distance euclidienne définie par la fonction suivante :

$$D((x_1, \dots, x_p), (u_1, \dots, u_p)) = \sqrt{(x_1 - u_1)^2 + \dots + (x_p - u_p)^2}$$

Le cas le plus simple de cet algorithme est lorsque k est égale à 1. 1-NN se base sur la classe du voisin le plus proche afin de classer l'instance inconnue.

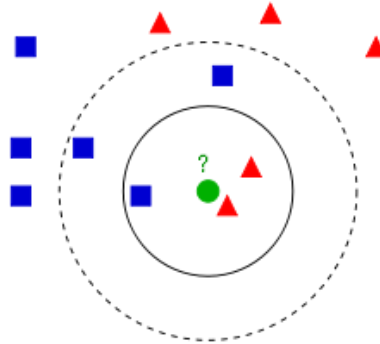
KNN se caractérise par la possibilité de faire une classification sans émettre d'hypomémoires sur la fonction reliant la variable dépendante (classe) aux variables indépendantes (instances), mais aussi, par l'influence de la valeur de k qui peut être choisie dans une échelle allant de quelques unités à quelques milliers, où les grandes valeurs de k produisent un lissage qui réduit le sur-apprentissage dû au bruit, ce qui est un avantage.

Un exemple de classification KNN est illustré par la figure 1 où le point inconnu (cercle)



appartient soit à la première classe (carré) ou bien la deuxième classe (triangle).

Si  $K = 3$ , le point inconnu est classé en deuxième classe parce qu'il y a deux triangles et un seul carré parmi les trois plus proches exemples à l'intérieur du cercle. Si  $K = 5$  il est classé dans la première classe (**Figure 1.7**). [12]



**Figure 1.7** : exemple de classification KNN.[12]

## 2.2 L'apprentissage non supervisé

Dans le cas de l'apprentissage non supervisé, on ne donne pas d'exemples de résultats attendus au système. Seules les données d'entrée sont fournies et le système doit apprendre, de façon autonome, la meilleure façon d'explorer les données. Il doit chercher à identifier dans le jeu de données une façon de les structurer (trouver des modèles ou « patterns ») ou encore à extraire des caractéristiques particulières. La performance permettant l'ajustement du modèle est alors appréciée grâce à des indicateurs objectifs, comme, par exemple, des calculs de variabilité intra ou interclasses [5].

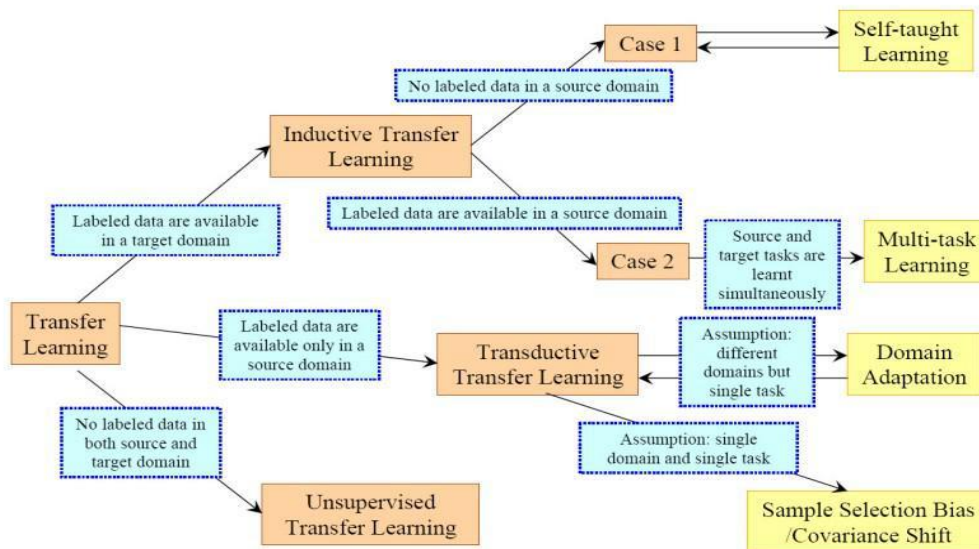
## 3 Transfer Learning

L'apprentissage par transfert est une méthode d'apprentissage machine dans laquelle un modèle développé pour une tâche est réutilisé comme point de départ pour un modèle sur une deuxième tâche.

C'est une approche populaire dans l'apprentissage approfondi où des modèles préformés sont utilisés comme point de départ pour des tâches de vision par ordinateur et de traitement du langage naturel, étant donné les vastes ressources de calcul et de temps nécessaires pour développer des modèles de réseaux neuronaux sur ces problèmes et les énormes sauts de compétence qu'ils permettent d'obtenir sur des problèmes connexes[13].

### 3.1 Les méthodes de transfer learning

Les domaines d'application du Transfer Learning sont nombreux. Principalement, les méthodes de transfert de connaissance sont très souvent utilisées pour la reconnaissance d'image ainsi que le traitement automatique du langage. Ces deux domaines d'apprentissage sont très complexes et chronophages. C'est pour cela que le Transfer Learning apporte un souffle nouveau pour tenter d'optimiser ces traitements en exploitant au maximum des modèles déjà entraînés. Nous allons voir dans la **figure 1.8** plusieurs types de Transfer Learning [14].



**Figure 1.8 :** Types de Transfer Learning.[14]

### 3.2 Apprentissage par transfert avec les données d'images

Il est courant d'effectuer un apprentissage par transfert avec des problèmes de modélisation prédictive qui utilisent des données d'image comme entrée [13]. Ils existent trois exemples de modèles qui attendent des données d'image comme entrée :

- Modèle Oxford VGG.
- Modèle de démarrage de Google.
- Modèle Microsoft ResNet.

## 4 Les réseaux de neurones artificiels et deep Learning

### 4.1 Les réseaux de neurones artificiels

Réseau de neurones ou réseau de neurones artificiels est l'un des mots à la mode les plus fréquemment utilisés en analytique de nos jours. Le réseau de neurone est une technique d'apprentissage automatique qui permet à un ordinateur d'apprendre des données d'observation. Le réseau de neurone en informatique est inspiré par la façon dont le système nerveux biologique traite les informations. Les réseaux de neurones biologiques sont constitués de neurones interconnectés avec des dendrites qui reçoivent des entrées. Sur la base de ces entrées, elles produisent une sortie via un axone vers un autre neurone [15].

C'est de ce mécanisme dont est inspiré le neurone artificiel, ou neurone formel, modélisé initialement par Warren McCulloch et Walter Pitts en 1943. Dans sa forme la plus simple, il peut se représenter de la façon suivante :

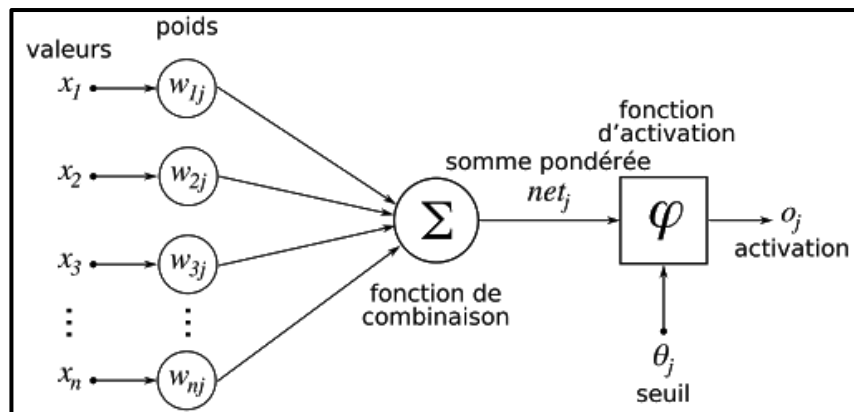
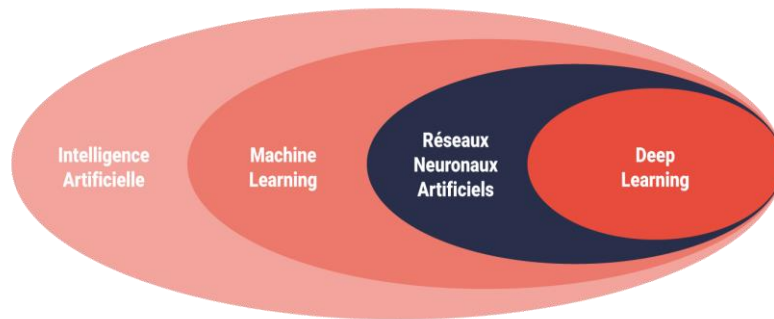


Figure 1.9 : un réseau de neurone artificiel [5].

Les valeurs ( $x_i$ ) sont les données d'entrée (les variables que l'on souhaite prendre en compte). On peut les comparer aux messages transmis par les liaisons entre les neurones (dendrites). Ces messages arrivent au neurone de manière pondérée c'est-à-dire que, pour traduire leur importance pour la résolution du problème, un poids ( $w_i$ ) leur est attribué (on parle de « poids synaptique »).

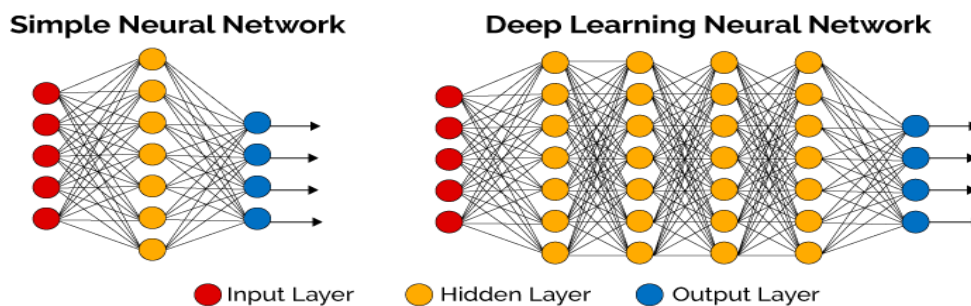
## 4.2 Deep Learning

Pour commencer, resituons le Deep Learning par rapport au vaste domaine de l'Intelligence Artificielle (IA). Le Deep Learning n'en est en effet qu'un sous ensemble. Il fait partie des réseaux de neuronaux artificiels qui sont une partie des méthodes d'apprentissage automatique (ou Machine Learning), constituant elles-mêmes l'une des branches de l'IA (**Figure 1.10**). [5]



**Figure 1.10** : Deep Learning.

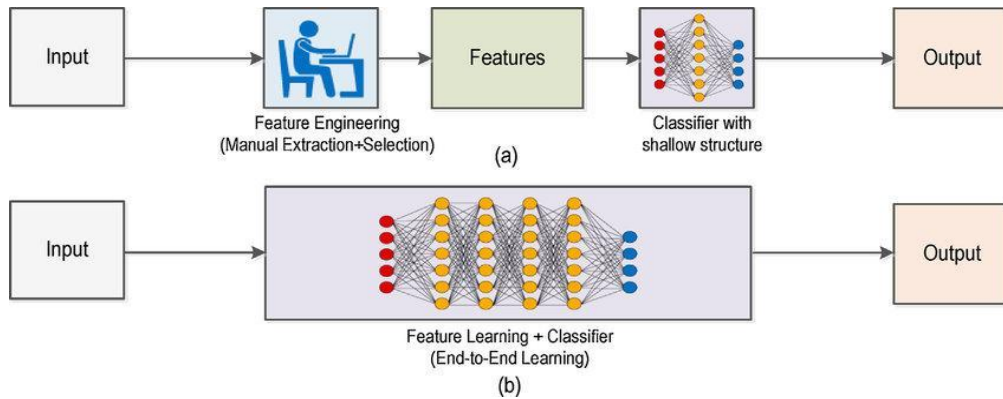
On utilise les mots « Deep » ou « profond » en référence au nombre de couches de neurones qui constituent ces réseaux : plus le nombre de couches est grand plus le réseau est profond et plus il permet de traiter des problèmes complexes. [5]



**Figure 1.11** : Des réseaux simples aux réseaux profonds. [5]

L'apprentissage profond évite la nécessité de sélectionner des fonctionnalités par un expert du domaine car il sélectionne automatiquement les fonctionnalités via la formation (**Figure 1.12**). Une autre caractéristique importante de l'apprentissage profond est qu'il a une capacité d'apprentissage considérablement plus élevée que les méthodes traditionnelles de ML, et qu'il peut donc apprendre des schémas très compliqués. En combinant ces deux caractéristiques, l'apprentissage profond est capable de l'apprentissage de la relation non

linéaire entre l'entrée brute et la sortie correspondante sans avoir à diviser le problème en petits sous-problèmes d'extraction et de classification des caractéristiques. Des travaux récents ont démontré l'efficacité des méthodes d'apprentissage profondi pour la classification (chiffrée) du trafic. Pour atteindre cet objectif, DL requiert données étiquetées suffisantes et puissance de calcul adéquate. [16]

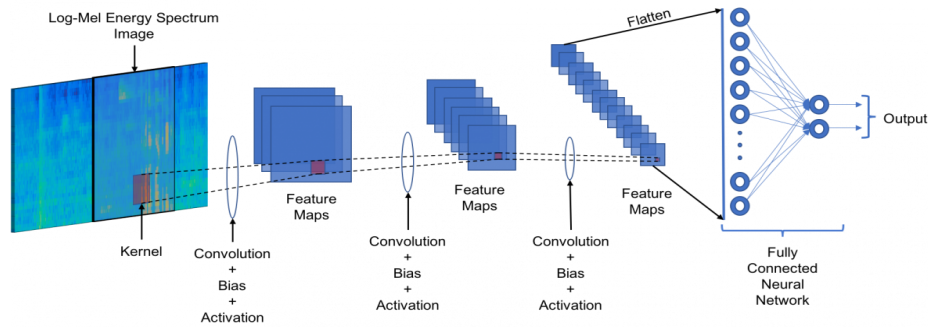


**Figure 1.12** : Comparaison entre l'apprentissage automatique (a) et Deep Learning (b). [5]

## 5 Les algorithmes de Deep Learning

### 5.1 Les réseaux de neurones convolutifs

Les réseaux de neurones convolutifs (CNN) sont biologiquement inspirés du cortex visuel de l'animal qui est composé d'une collection de cellules (également appelées neurones) connectées les unes aux autres et organisées en couches hiérarchiques. Chaque ensemble de neurones répond très spécifiquement à des schémas spécifiques (également appelés champ récepteur de la cellule). En général, toutes les architectures CNN se composent d'un ensemble de couches. La première s'appelle la couche convolutionnelle suivie de l'activation et de la couche mise en commun, ces trois couches forment une couche cachée. La dernière couche est la couche dense (appelée couche entièrement connectée). L'architecture du CNN est illustrée à la **figure 1.13**. [17]



**Figure 1.13** : Architecture de réseau neuronal convolutif [17].

### ✚ Notation

Nous désignons  $x^i$  comme une seule valeur d'entité dans la carte d'entités. Nous définissons également  $x^i$  comme une valeur de probabilité unique dans la carte d'entités normalisée  $\mathbf{W}$ . Cette normalisation est effectuée en appliquant la fonction softmax à chaque  $x^i$  de la carte d'entités. Nous représentons la perte résultante de la fonction de perte d'entropie croisée sur la carte des entités  $\mathbf{W}$  comme  $\mathbf{J}(\mathbf{W})$  et nous notons  $\eta \times \partial \mathbf{j}(\mathbf{W}) / \partial \mathbf{w}$  comme la dérivée partielle de la fonction de perte (également appelée gradient), multipliée par  $\eta$  où  $\eta$  représente la valeur du taux d'apprentissage. Nous désignons également la sortie oracle du classificateur comme  $y_i$  qui est égal à 1 pour représenter la présence de la classe et 0 sinon.

### ✚ La couche convolutionnelle

Il se compose d'un ensemble de filtres (également appelé matrice d'entités, noyau, poids et carte d'entités). Le filtre se déplace sur chaque partie de l'entrée avec une certaine valeur de pas et une opération mathématique appelée convolution est calculée. Il se compose d'un produit scalaire entre la matrice qui représente la partie  $p$  de l'entrée et la matrice de carte d'entités. Ce processus est répété jusqu'à ce que l'entrée entière soit traversée. La matrice résultante est connue sous le nom de matrice d'entités convolutée (ou simplement carte d'entités), elle se compose d'un ensemble de valeurs d'entités qui correspondent exactement à l'entrée et d'un ensemble de zéros qui représentent les parties non pertinentes où il n'y avait pas de correspondance. L'objectif de la convolution est de permettre au modèle CNN d'apprendre uniquement les fonctionnalités les plus pertinentes [18]. Une illustration de cette opération est illustrée à la **figure 1.14**.

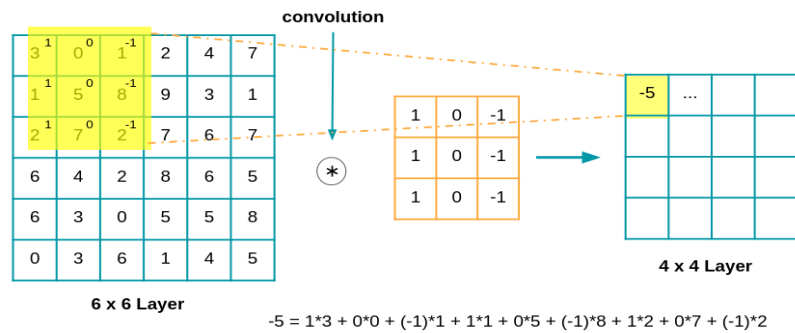


Figure 1.14 : Convolution dans CNN. [18]

### 🚦 Couche d'activation

La couche d'activation applique une fonction d'activation sur toute la matrice d'entités convolutées pour permettre au modèle d'apprendre la fonction de cartographie non linéaire  $f$  pour classer les données non linéaires. La fonction d'activation la plus couramment utilisée connue sous le nom de fonction d'activation des unités linéaires rectifiées (Relu). Il est défini comme suit : [19]

$$RELU(x) = \begin{cases} 0 & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases} \quad (8)$$

### 🚦 Couche pooling

La couche Pooling est visible entre les couches Convolution dans une architecture CNN. Cette couche réduit fondamentalement le nombre de paramètres et de calcul dans le réseau, contrôlant le sur-ajustement en réduisant progressivement la taille spatiale du réseau. Il y a deux opérations dans cette couche; Pooling moyen et pool maximum. Seul Max-pooling sera discuté dans ce poste (**Figure 1.15**).

#### ➤ Max-pooling :

Comme le nom l'indique; ne retirera que le maximum d'une piscine. Cela se fait en fait avec l'utilisation de filtres glissant à travers l'entrée; et à chaque foulée, le paramètre maximum est supprimé et le reste est supprimé. Cela sous-échantillonne en fait le réseau.

Contrairement à la couche de convolution, la couche de mise en commun ne modifie pas la profondeur du réseau, la dimension de profondeur reste inchangée. [20]

**Formule pour la sortie après Max-pooling : [48]**

$$y = \frac{(N-F)}{S_1} \tag{9}$$

- N = dimension de l'entrée dans la couche de regroupement
- F = Dimension du filtre
- S = foulée

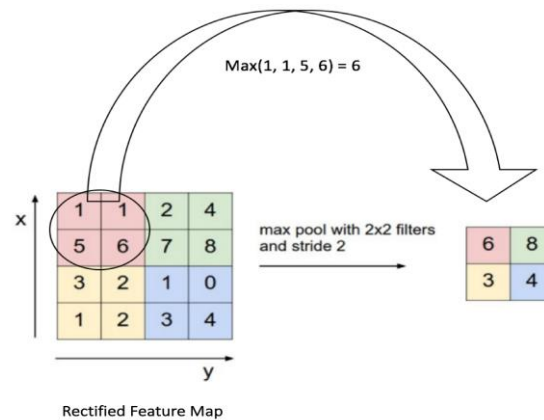


Figure 1.15 : pooling. [18]

### 🚦 Couche dense

Les couches convolutionnelle et max\_pooling produisent une carte d'entités. La couche dense qui est la dernière couche du CNN aplatit cette matrice et la normalise en utilisant une fonction softmax en un vecteur de valeurs de probabilité  $\in [0,1]$  dont le total est égal à 1. Ce vecteur est ensuite utilisé pour former le CNN en afin d'effectuer la classification. L'entraînement se fait en trouvant les poids (carte des caractéristiques) qui minimisent la perte entre la sortie réelle et prédite dans l'ensemble d'apprentissage en utilisant la fonction de perte d'entropie croisée. Ce processus est connu sous le nom de descente de gradient (également appelé rétropropagation) qui sera répété sur une série d'époques (itérations) pour mettre à jour la carte des entités jusqu'à la convergence. Après l'étape d'apprentissage, la couche dense génère une distribution de probabilité pour chaque classe prédéfinie. La formule mathématique de la fonction softmax est présentée ci-dessous:

$$softmax(Z_j) = \frac{e^{Z_{j1}}}{\sum_{K=1}^K e^{Z_K}} \text{ for } j = 1, \dots, K \tag{10}$$

### 🚦 Gradient descent :

La descente en gradient est le processus de recherche des poids qui minimisent la fonction de perte sur une série d'époques pendant la phase d'apprentissage jusqu'à la convergence. On



dit que cet algorithme d'optimisation converge lorsque, au fur et à mesure des itérations sur une série d'époques, le gradient se rapproche de 0 et reste stable.

Afin de mesurer les performances d'un modèle de classification qui produit des sorties oracle, nous utilisons la fonction de perte d'entropie croisée binaire. L'algorithme Gradient Descent et les formules mathématiques d'entropie croisée binaire sont présentés ci-dessous :

$$j(W) = \frac{1}{n} \sum_{i=1}^n (y_i \log(x_i) + (1 - y_i) \log(1 - y_i)) \quad (11)$$

### 5.1 Réseau neuronal résiduel (ResNet)

Residual Network (ResNet) est une architecture de réseau neuronal convolutif (CNN) qui peut prendre en charge des centaines de couches convolutives ou plus. ResNet peut ajouter de nombreuses couches avec de fortes performances, alors que les architectures précédentes avaient une baisse d'efficacité avec chaque couche supplémentaire.

ResNet a proposé une solution au problème du "gradient de disparition". Les réseaux neuronaux s'entraînent par rétropropagation, qui repose sur la descente du gradient pour trouver les poids optimaux qui minimisent la fonction de perte. Lorsque des couches supplémentaires sont ajoutées, la multiplication répétée de leurs dérivés finit par rendre le gradient infiniment petit, ce qui signifie que des couches supplémentaires n'amélioreront pas les performances ou peuvent même les réduire (**Figure 1.16**).

ResNet résout ce problème en utilisant des "connexions de raccourcis d'identité" - des couches qui, au départ, ne font rien. Dans le processus de formation, ces couches identiques sont sautées, réutilisant les fonctions d'activation des couches précédentes. Le réseau est ainsi réduit à quelques couches seulement, ce qui accélère l'apprentissage. Lorsque le réseau s'entraîne à nouveau, les couches identiques s'étendent et aident le réseau à explorer davantage l'espace des fonctionnalités [21].

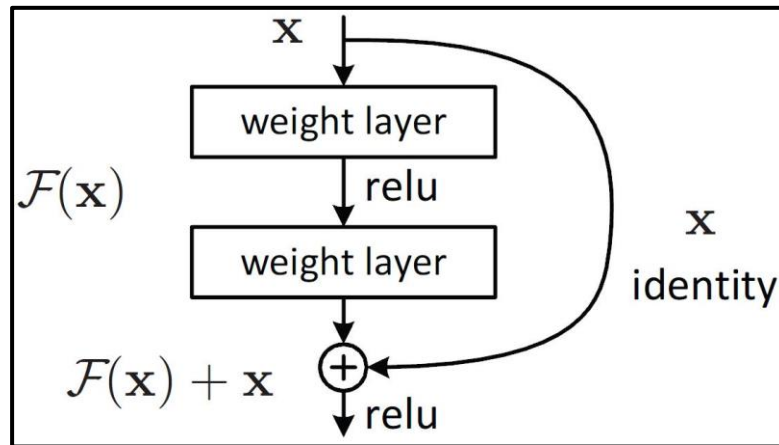


Figure 1.16: La formule de ResNet.

## 6 Cycle de vie de l'implémentation des algorithmes

### 6.1 Prétraitement

Comme le modèle est construit à partir des données, il est clair que plus on dispose de données, plus le modèle construit est précis et permettra ainsi de prendre de bonne décisions. Mais Quand les données d'un DataSet sont dans des ordres de grandeurs différents, certains algorithmes de Machine Learning et Deep Learning mettent plus de temps à trouver un modèle prédictif optimal. [22]

Le prétraitement et le nettoyage des données sont alors des tâches importantes qui doivent intervenir avant d'utiliser un jeu de données à des fins d'apprentissage automatique ou profond. Les données brutes sont souvent bruyantes, peu fiables et incomplètes. Leur utilisation pour la modélisation peut générer des résultats trompeurs. [23]

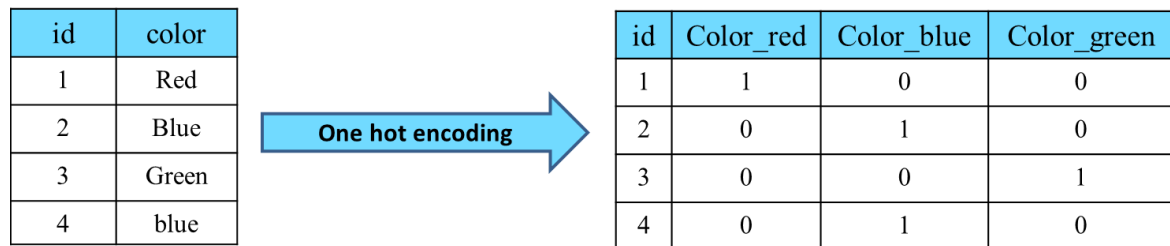
La première étape à réaliser est donc l'obtention de données en suffisance, représentatives du problème à résoudre. En effet, s'il s'avère qu'on a beaucoup de données d'entraînement et/ou que l'algorithme d'apprentissage est lourd, il est possible d'utiliser toutes les données prenne énormément de temps et/ou de ressources hardware. Dans ce cas, il faut naturellement échantillonner et ne récupérer qu'un petit pourcentage du dataset qui servira au travail de modélisation pour aller plus vite. Le problème lorsqu'on effectue un échantillonnage, c'est que l'on doit être bien sûr que cet échantillon est représentatif de toutes les données. On parle d'étape de sampling en anglais. [24]

De nombreux algorithmes d'apprentissage machine ne peuvent pas fonctionner directement sur des données tabulaires. Ils exigent que toutes les variables d'entrée et de sortie soient numériques. Cela signifie que les données catégorielles doivent être converties sous une forme numérique. Pour convertir les données catégorielles en données numériques [25],

cela implique deux méthodes :

### ✚ One Hot Encoding

Le One Hot Encoding est une méthode courante de prétraitement des caractéristiques catégorielles pour les modèles d'apprentissage machine. Ce type de codage crée une nouvelle caractéristique binaire pour chaque catégorie possible et attribue une valeur de 1 à la caractéristique de chaque échantillon qui correspond à sa catégorie d'origine. C'est plus facile à comprendre visuellement : dans **Figure 1.17** ci-dessous, nous codons une caractéristique de couleur qui se compose de trois catégories (rouge, vert et bleu) [26].



**Figure 1.17** : one hot encoding.

Nous appliquons cette méthode lorsque :

- Le nombre de caractéristiques catégorielles est moindre, ce qui permet d'appliquer efficacement l'encodage en une seule fois. [27]

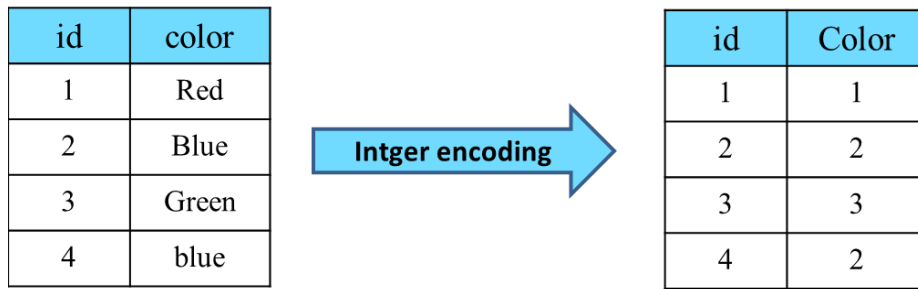
### ✚ Integer Encoding

Dans cette méthode, chaque valeur de catégorie unique se voit attribuer une valeur entière. Par exemple (**Figure 1.18**), "rouge" est 1, "vert" est 2 et "bleu" est 3.

Les valeurs entières ont une relation ordonnée naturelle entre elles et les algorithmes d'apprentissage peuvent être capables de comprendre et d'exploiter cette relation. [25]

Nous appliquons cette méthode lorsque :

- Le nombre de catégories est assez large car le codage en une seule fois peut entraîner une forte consommation de mémoire. [27]



**Figure 1.18 :** Integer encoding.

La deuxième étape est le nettoyage, appelé aussi << pré-processing >> , de la donnée récoltée, cette étape consiste en un prétraitement de toutes ces données avant de commencer à les analyser : Gestion des valeurs manquantes, la suppression des datas aberrantes, mise sous forme tabulaire des données ; le but de cette étape est de rendre la modélisation plus simple et rapide, d'optimiser le temps d'exécution et d'apprentissage mais aussi d'extraire les caractéristiques des données pertinentes pour la prise de décision autrement appelées caractéristiques (features). Cette extraction de caractéristiques (features sélection) est une bonne pratique, pour ne pas dire obligatoire, lors de la modélisation avec du Machine Learning. [28]

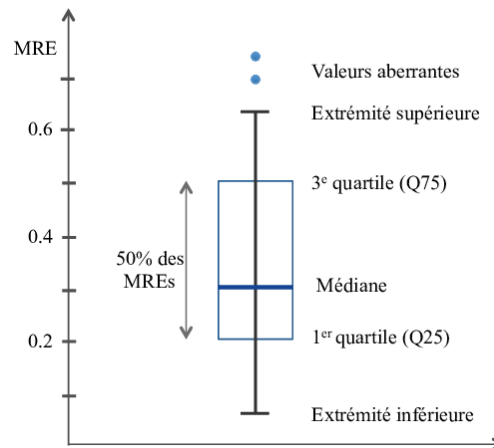
#### Gestion des valeurs manquantes

Les données manquantes sont un problème courant dans l'analyse pratique des données. Dans les ensembles de données, les valeurs manquantes peuvent être représentées par " ? ", " nan ", " N/A ", une cellule vide, ou parfois par " -999 ", " inf ", " -inf ".

Le mécanisme des données manquantes peut être classé en trois types : données manquantes complètement au hasard (MCAR), données manquantes au hasard (MAR) et données manquantes non au hasard (MNAR). De manière informelle, MCAR signifie que l'occurrence des valeurs manquantes est complètement aléatoire, sans lien avec aucune variable. MAR implique que les valeurs manquantes ne concernent que les données observées et MNAR désigne le cas où les valeurs manquantes sont liées à la fois à la variable observée et à la variable non observée et où le mécanisme manquant ne peut être ignoré. [29]

### ✚ Les données aberrantes

Les valeurs aberrantes sont des points de données qui sont loin des autres points de données (**Figure 1.19**). En d'autres termes, ce sont des valeurs inhabituelles dans un ensemble de données. Les valeurs aberrantes sont problématiques pour de nombreuses analyses statistiques, car elles peuvent faire en sorte que les tests manquent des résultats significatifs ou faussent les résultats réels. [30]



**Figure 1.19** : Exemple de valeurs aberrantes.

### ✚ Features selection

Les méthodes de sélection des caractéristiques visent à réduire le nombre de variables d'entrée à celles qui sont considérées comme les plus utiles à un modèle afin de prédire la variable cible.[31]

Les processus de sélection, d'un ou plusieurs sous-ensembles optimaux à partir de l'ensemble formé par tous les attributs, se fait par trois types d'approches : wrapper [32], filter [33] et embedded [34].

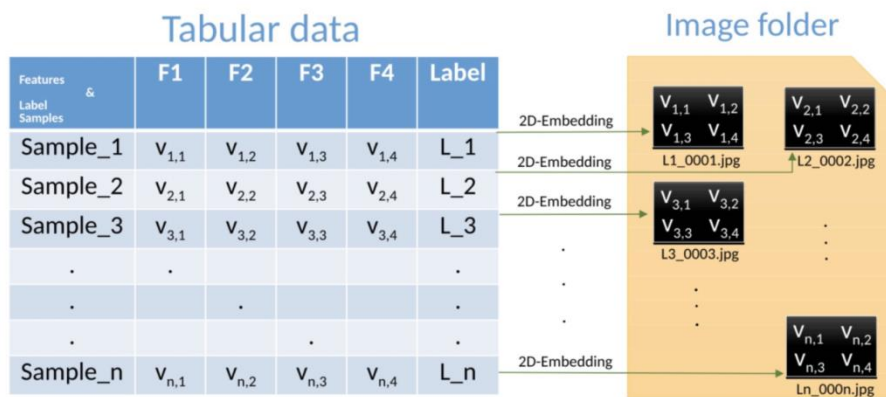
Les méthodes « wrappers » utilisent les algorithmes de classification pour générer et ensuite évaluer la qualité des ensembles candidats. Cette approche est généralement pertinente mais elle dépend de la représentativité de l'ensemble d'apprentissage.

Les méthodes « filters » sont basées sur une fonction critère pour mesurer la pertinence de l'information contenue dans les ensembles candidats. Elles réduisent considérablement le temps de calcul alors que les méthodes « wrappers » permettent d'obtenir des meilleurs résultats.

Les méthodes « embedded » tentent de combiner les avantages de précédentes approches. Elles intègrent la phase de sélection des variables dans l'étape d'apprentissage. Néanmoins, le temps de calcul reste important. C'est probablement la raison principale pour laquelle les méthodes « filters » sont les plus populaires.

Certains algorithmes du deep learning ne peuvent pas classifier les données tabulaires. Afin qu'ils puissent faire la classification, il est nécessaire de les convertir en image par une méthode qui s'appelle SuperTML (**Figure 1.20**). Cette dernière fonctionne comme suit :

1. Créer un encastrement bidimensionnel (two-dimensional embedding en anglais).
2. Projeter les caractéristiques des données tabulaires sur les images générées.



**Figure 1.20** : Un exemple de conversion de données tabulaires en images avec two-dimensional embedding. [35]

## 6.2 La modélisation

Une fois les données prétraitées, il va s'agir de trouver le bon algorithme. De nombreux choix d'algorithmes d'apprentissage et de leurs hyper paramètres s'offrent aux Data Scientists. La nature du problème à résoudre permet en partie de guider ce choix. Ce choix doit donc être fait en fonction des résultats désirés ainsi que de la complétude des données [36].

Les facteurs qui peuvent entrer en jeu pour choisir le bon algorithme peuvent être nombreux, notamment le nombre de caractéristiques (features), la quantité de données qu'on a...etc. [37]

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

régression, le second une classification.

Néanmoins, il est nécessaire de savoir évaluer n'importe quel algorithme d'apprentissage sur son jeu de données. Une évaluation rigoureuse des performances d'un algorithme est une étape indispensable à son déploiement [36].

### 6.3 La phase d'apprentissage

La phase d'apprentissage repose sur un algorithme d'apprentissage c'est-à-dire la mise en place d'une architecture et d'un programme informatiques qui permettent à une machine de recevoir des données d'entrée, d'effectuer une série de traitements utilisant ces données d'entrée, de produire un résultat en sortie et, surtout, de s'améliorer pour produire ce résultat. Ce dernier point suppose qu'un objectif de résultat (et éventuellement une tolérance par rapport à l'atteinte de cet objectif) soit communiqué au système et que des données d'entraînement et les moyens de mesurer ses performances lui soient fournis. C'est en cherchant à améliorer ses performances sur les données d'entraînement que la machine va apprendre.

Une fois que l'apprentissage sera terminé, la « machine » pourra produire le résultat de façon autonome sur des données d'entrée qu'elle n'aura encore jamais rencontrées.

Mais, selon le résultat visé, les données dont on dispose au départ et le contexte, il existe plusieurs façons de faire apprendre à une machine. On distingue notamment l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement (d'autres méthodes existent mais nous ne les développerons pas ici).

### 6.4 Validation

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

### **Hold-out :**

Hold-out ou «la rétention» consiste à diviser votre ensemble de données en un ensemble «train» et «test». L'ensemble d'apprentissage est ce sur quoi le modèle est formé, et l'ensemble de test est utilisé pour voir dans quelle mesure ce modèle fonctionne sur des données invisibles. Une division commune lors de l'utilisation de la méthode de maintien consiste à utiliser 80% des données pour la formation et les 20% restants des données pour les tests. [38]

### **Cross-validation :**

Cross-validation ou «validation croisée k-fold» signifie que le jeu de données est divisé au hasard en groupes «k». L'un des groupes est utilisé comme ensemble de test et les autres sont utilisés comme ensemble de formation. Le modèle est formé sur l'ensemble d'entraînement et noté sur l'ensemble d'essai. Ensuite, le processus est répété jusqu'à ce que chaque groupe unique soit utilisé comme ensemble de test [38].

## 6.5 Performance du modèle

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

On mesure la performance du modèle sur la base de test. Il existe certaine façon standard de le faire en fonction des types de problèmes : [39]

### **Classification :**

- Matrice de confusion.
- Le rapport de classification (Précision, rappel et f\_score).

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

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

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



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

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

## 7 Conclusion

À travers ce chapitre, nous avons passé en revue les concepts importants de la classification. Tout d'abord, nous avons présentés la machine Learning et le transfer learning, en passant brièvement en revue les réseaux de neurones artificiels et Deep Learning. Ensuite, nous avons spécifiés les algorithmes d'apprentissage utilisés pour notre travail. Finalement nous avons parlé sur le cycle de vie d'implémentation de machine Learning.

Le chapitre suivant sera consacré à la deuxième phase de notre projet à savoir la classification du trafic chiffré ainsi que le trafic VPN et les travaux connexes sur cette approche.

# Chapitre 2: Classification Du Trafic Réseau Chiffré

## 1 Introduction

La classification du trafic réseau est une tâche importante dans les réseaux de communication modernes [1]. Par conséquent, la classification précise du trafic est devenue l'une des conditions préalables aux tâches avancées de gestion de réseau telles que la fourniture d'une qualité de service (QoS) appropriée, la détection des anomalies, la tarification, etc.

L'émergence de nouvelles applications ainsi que les interactions entre les différents composants de l'Internet ont considérablement accru la complexité et la diversité de ce réseau, ce qui rend la classification du trafic problème difficile en soi.

Ce chapitre est organisé comme suit: La section 2 présente la définition de trafic réseau ainsi que les types de trafic réseau. La section 3 la définition de flux de donnée. La section 4 traite la classification du trafic réseau et aussi les méthodes de la classification du trafic, en suite dans La section 5 on a décrire la définition d'un VPN et à la fin dans la section 6, nous avons présentés les travaux connexes sur ce domaine. Le chapitre est conclu à la section 7.

## 2 Trafic réseau

Le trafic réseau fait référence à la quantité de données se déplaçant sur un réseau à un moment donné. Les données du réseau sont principalement encapsulées dans des paquets réseau, qui fournissent la charge dans le réseau. Le trafic réseau est le composant principal pour la mesure du réseau, le contrôle du trafic réseau et la simulation. La bonne organisation du trafic réseau contribue à garantir la qualité de service dans un réseau donné. [40]

## 2.1 Paquet réseau

Le paquet est l'unité de transmission de données qui est acheminée entre une origine et une destination sur un réseau pour établir une communication.

Afin de transmettre un message d'une machine à une autre sur un réseau, celui-ci est découpé en plusieurs paquets transmis séparément.

Lorsqu'un fichier est envoyé d'un point à un autre sur Internet, la couche TCP (Transmission Control Protocol) du Protocol TCP/IP le scinde en « fragments » de taille adaptée pour le routage. Chacun de ces paquets, numérotés séparément, comprend l'adresse IP de la destination (**Figure 2.1**).

Les paquets individuels d'un fichier donné peuvent suivre des parcours différents sur Internet. Une fois tous parvenus à destination, ils sont réassemblés pour reformer le fichier d'origine (au niveau de la couche TCP côté réception).

Un paquet inclut les données, encapsulées dans un "en-tête", comprenant des informations utiles pour transporter et décoder le message. Exemple : paquet IP...

Le paquet est lié à un niveau 3 dans le modèle OSI. Si l'on parle de trame c'est à la couche 2 du modèle OSI que l'on fait référence. Exemple : Ethernet

Un système à commutation de paquets permet de gérer efficacement les transmissions sur un réseau en mode sans connexion tel qu'Internet. Les termes « paquet » et « datagramme » ont la même signification. [41]

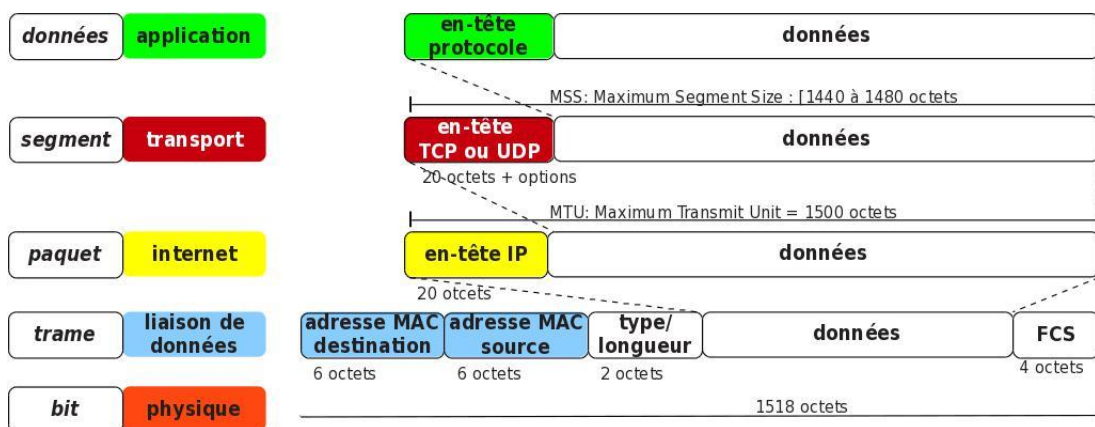


Figure 2.1 : Réseaux de données [41].

### 3 Flux de donnée

Lorsque les ordinateurs ont besoin de se parler, ils établissent des canaux de communication, communément appelés connexions. (Techniquement parlant, ces canaux de communication ne peuvent être appelés connexions que lorsque le protocole TCP est impliqué). Un flux désigne toute connexion ou tout canal de communication de type connexion.

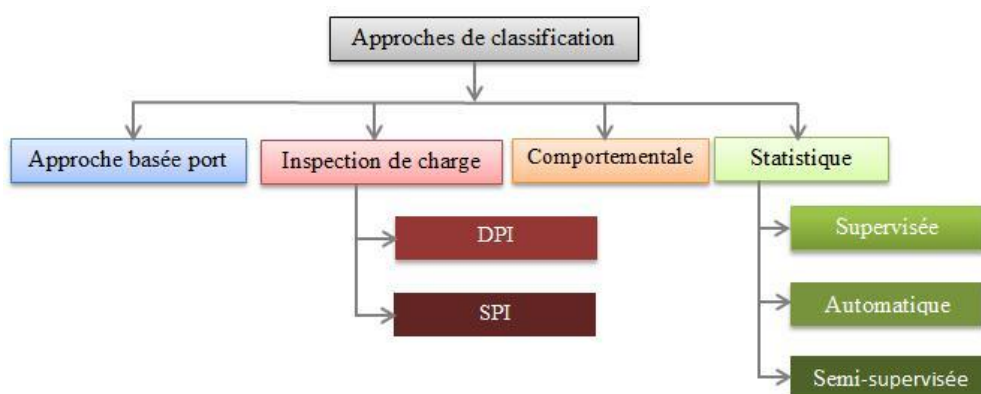
En termes plus techniques, un flux est défini par son 5-tuple, un ensemble de cinq points de données:

Adresses IP source et destination échangeant des informations Ports source et destination, le cas échéant (ICMP, par exemple, n'utilise pas de ports) Le protocole Un flux identifie un canal de communication, et tous les paquets partageant les mêmes champs à 5 tubes appartiennent au même flux [42].

### 4 Classification du trafic réseau

La classification des flux réseau en fonction du type d'application est la technique de base pour de nombreux systèmes de surveillance et de contrôle réseau, notamment, les systèmes de gestion de la qualité de service et les systèmes de sécurité (détection d'anomalies et d'intrusions). Nous utilisons cette approche pour décrire les méthodes de classification du trafic en fonction des caractéristiques observées passivement dans le trafic et selon des objectifs de classification spécifiques.

Les approches de la classification du trafic sont représentées dans la **figure 2.2** ci-dessous :



**Figure 2.2** : Les approches de la classification.

#### 4.1 Classification basée sur les ports

Les premières solutions de classification de trafic se basent principalement sur les numéros de ports pour classer les applications [43]. Celles-ci sont connues dans la littérature sous le nom de la classification basée sur les ports. Cette approche est particulièrement simple et rapide en comparaison avec d'autres approches, elle classe efficacement les applications standards vu qu'elles utilisent des ports assignés par l'IANA qui sont bien connus. Bien qu'elle présente des avantages, la classification basée sur les ports est devenue inefficace avec la présence des applications non standards telle que le trafic P2P (peer to peer). En fait, ces dernières peuvent contourner les systèmes de contrôle d'accès de plusieurs façons, par exemple en utilisant des ports non enregistrés ou via une allocation dynamique ou encore en utilisant les ports standards et enregistrés des autres applications.

#### 4.2 Classification par l'inspection de charge

Pour pallier à la limite de la classification précédente, une approche alternative, dite la classification par l'inspection de charge des paquets, ou (Payload based Classification) a été envisagée. Cette approche consiste à examiner la charge utile de chaque paquet dans sa recherche d'un indice ou la signature de l'application. Elle se décline en deux branches ; i) Inspection profonde de paquets (DPI pour Deep Packet Inspection) et ii) inspection stochastique de paquets (SPI pour Stochastic Packet Inspection).

La première branche repose sur une inspection mécanique de la charge utile de chaque paquet lors de la recherche d'une expression ou d'un mot clé caractérisant une application donnée. Moore et Papagiannaki ont proposé un classificateur comportant deux étages combinant l'approche basée sur les ports et la classification par l'inspection de charge [44].

Le premier étage examine le numéro de port, et lorsque le flux utilise un port non standard, le second étage se charge de chercher dans les premiers octets de la charge utile du premier paquet une signature ou un protocole connu. Une inspection détaillée de toute la charge utile est nécessaire pour les flux qui restent non classifiés. Les résultats décrits dans cette publication démontrèrent que le premier étage permet de classer correctement 69 % des octets en utilisant uniquement le numéro de port. Ce taux augmente pour atteindre 79 % en incluant les informations issues des premiers octets du premier paquet. La dernière étape, qui examine les flux non classifiés, permet d'obtenir un résultat élevé, à savoir de presque 100 %. L'Approche discutée dans [45] démontre que la classification du trafic P2P, par l'inspection de la charge, permet de réduire le taux des faux positifs et faux négatifs jusqu'à 5

% du total des octets. Cette approche présente des avantages, notamment son taux de classification élevé. De plus, elle est susceptible d'être utilisée dans un processus de classification de trafic en ligne car la signature peut être déduite à partir des premiers paquets des flux. Pour ces raisons, cette approche est implémentée dans plusieurs solutions, telles que la détection d'intrusion [46] et le pare-feu de Linux (L7-filter). Néanmoins, elle souffre de plusieurs problèmes, en particulier celui du trafic chiffré.

La deuxième famille de techniques reprend le même principe de base qui est l'inspection de la charge, mais d'une manière statistique de façon à chercher les propriétés distinctives de chaque application. Elle vise à combler certaines lacunes de la première famille de techniques. Ainsi, elle utilise des méthodes automatiques pour former des modèles distinctifs. Plusieurs techniques de reconnaissance de forme sont proposées. Par exemple, les auteurs [47] ont utilisé la valeur des premiers octets de la charge utile comme entrée pour leurs algorithmes d'apprentissage machine (c.-à-d. Naive Bayes, Adaboost et Maximum Entropy) pour classifier les applications. Les auteurs de [48] ont développé un classifieur pour distinguer les types des contenus des paquets en utilisant l'entropie des premiers octets de la charge et les techniques d'apprentissage automatique. En effet, l'entropie constitue une caractéristique distinctive du fait qu'elle dépend de la nature du contenu du paquet ; les valeurs d'entropie les plus petites correspondent à un texte simple, les moyennes distinguent un contenu binaire et les plus grandes correspondent à un contenu chiffré.

Bien que les méthodes SPI (Stochastic Packet Inspection) permettent de distinguer la nature de plusieurs types de contenu de trafic, y compris le contenu chiffré, ce qui peut être utile pour prioriser un trafic par rapport aux autres [48]. Ces méthodes héritent de plusieurs problèmes des méthodes DPI (Deep Packet Inspection) et elles sont incapables d'identifier le type du trafic du fait que certaines applications peuvent utiliser plusieurs types de contenu en même temps.

### **4.3 Approche comportementale**

L'approche comportementale se focalise sur l'analyse du comportement des hôtes et de la distribution des connexions pour déduire le type de trafic avec le but d'identifier les applications actives sur un hôte donné. Les classificateurs de cette catégorie examinent les patrons générés par le trafic en observant un certain nombre de paramètres, tel que le nombre d'hôtes qui y sont connectés, le nombre de ports et les protocoles utilisés. L'idée derrière une telle approche est que différentes applications génèrent des patrons différents. Par exemple un serveur Web est interrogé par plusieurs clients par des sockets parallèles tandis que dans un

réseau P2P (Peer to Peer) les hôtes sont interconnectés avec le même degré de popularité. L'outil BLINC [49] est une solution qui permet de classer le trafic en se référant à l'allure des communications au niveau d'un hôte. Les concepteurs de TDG [50] et [51] ont utilisé les répartitions graphiques des flux ainsi que des métriques de classification (degré de connexion, I/O, etc.) pour classer le trafic.

### 4.4 Approche statistique

La classification du trafic par l'approche statistique s'appuie d'un côté sur les techniques d'apprentissage machine et d'un autre côté sur le fait que les différents types de trafic possèdent différentes caractéristiques ou métadonnées telles que la taille des paquets (petite ou grande), la taille des flux, le temps inter arrivé des paquets, la durée des flux, etc. En effet, chaque flux est décrit par un vecteur de caractéristiques qui réfère à une observation ou une instance dans le jeu de données [52].

Dans notre travail nous nous sommes orientés vers l'approche statistique. On a utilisé divers algorithmes d'apprentissage, pour obtenir une classification d'un trafic réseau vpn et non VPN en fonction de type d'application.

## 5 Définition d'un VPN

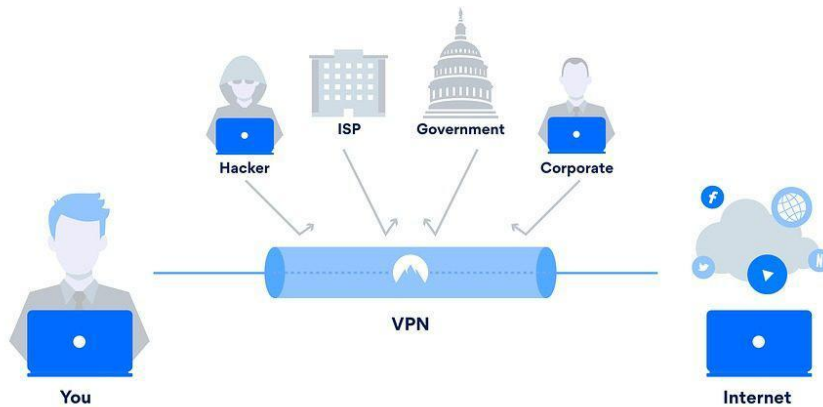
VPN signifie Virtual Private Network (réseau privé virtuel). Ce réseau privé vous permet d'envoyer votre trafic de données via une connexion chiffrée et sécurisée à un serveur externe (**figure 2.3**). De là, le trafic sera envoyé sur l'internet. De ce fait, l'adresse IP indiquée sur l'internet sera modifiée.



**Figure 2.3** : un VPN [53].

Autrement un VPN offre une sécurité car il crypte fortement tout votre trafic Internet, avant même qu'il n'atteigne le serveur VPN. Il guide également votre trafic de données à

travers un "tunnel VPN" beaucoup plus sûr. Il est donc beaucoup plus difficile pour d'autres personnes, comme les gouvernements et les pirates informatiques, d'intercepter et de visualiser vos données. Il est donc particulièrement important d'utiliser un VPN lorsque vous utilisez des réseaux WiFi publics (dangereux). La **figure 2.4** exprime ce qu'on parle ci-dessus



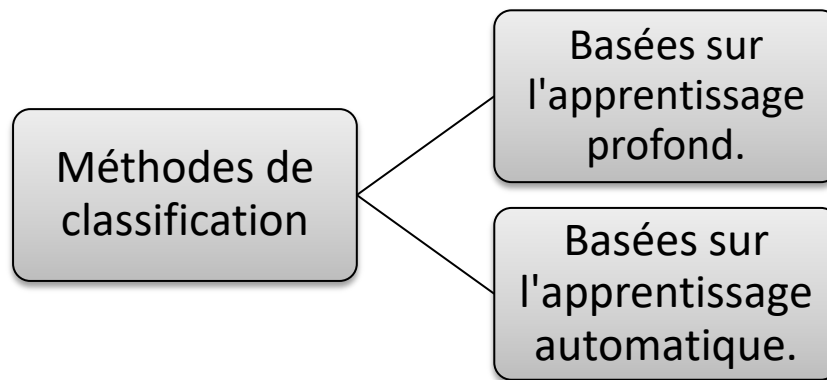
**Figure 2.4** : Une communication VPN [53].

Enfin, un VPN offre l'anonymat et la protection de la vie privée en ligne pour cela un VPN offre plus d'anonymat en ligne car vous ne naviguez pas sur le web avec votre adresse IP accessible au public. Votre adresse IP personnelle sera cachée dès que vous vous connecterez au serveur VPN, car elle deviendra l'adresse IP du serveur VPN. Normalement, d'autres personnes peuvent établir un lien entre vos actions en ligne et votre identité et votre localisation en se basant sur votre adresse IP. Par exemple, votre fournisseur d'accès internet, les sites web que vous visitez et de nombreux gouvernements peuvent généralement voir tout ce que vous faites en ligne. [53]

## 6 Les travaux connexes

Les recherches actuelles sur la classification se concentrent principalement sur l'approche statistique [54]. De nombreuses recherches sont actuellement menées sur la classification basée sur les classificateurs d'apprentissage automatique et profond.





**Figure 2.5 :** Méthodes de classification.

La première catégorie se concentre principalement sur les méthodes classiques de machine learning par. Par exemple, Wang et al. [52], Coull et al. [55] et Mauro et al. [56] ont fait des recherches sur le P2P, iMessage et WebRTC respectivement. Ils ont appliqué respectivement les caractéristiques de flux et les caractéristiques de paquets. Les classificateurs correspondants sont l'arbre de décision C4.5, l'arbre de décision Naive respectivement Bayes et Random Forest. Relativement parlant, moins de recherches sont menées sur la classification du trafic encapsulé dans les protocoles. Par exemple, Aghaei et autres [57] ont proposé une méthode de classification avec des caractéristiques de flux et un classificateur d'arbre de décision C4.5 sur le trafic proxy. Draper-Gil et al. [58] ont proposé une méthode de classification avec uniquement des caractéristiques de flux liées au temps à la fois sur le trafic chiffré normal et sur le trafic encapsulé dans le protocole. Il est à noter qu'ils ont publié un ensemble de données très utile incluant ces deux types de trafic.

**Tableau 2.1 :** Travaux liés à la classification machine learning.

Recherche	Année	Datasets	Caractéristiques	Classificateur	Performances
Wang [52]	2014	P2PMessage	Les caractéristiques de flux	Arbre de décision (c.4.5)	94%
Coull [55]	5 Oct 2014	WebRTC	Les caractéristiques de paquet	Naive de bays	96%
Mauro [56]	2015	WebRTC	Les caractéristiques de flux	Random forest	97%
Aghaei [57]	2016	Le trafic proxy	Les caractéristiques de flux	Arbre de décision (c.4.5)	98%
Draper-gil [58]	2015	Le trafic chiffré normal et le trafic	Les caractéristiques de flux liées au temps à la fois	Arbre de décision (c.4.5)	80%

encapsulé  
sur le  
Protocol

La deuxième catégorie se concentre principalement sur les méthodes d'apprentissage profond. Par exemple, mohammad lotfollahi et al. [15] ont proposé une approche basée sur l'apprentissage approfondi qui intègre à la fois les phases d'extraction de caractéristiques et de classification dans un seul système. Le schéma qu'ils ont proposé, appelé "Deep Packet", peut gérer à la fois la caractérisation du trafic, dans laquelle le trafic réseau est classé en grandes classes (par exemple, FTP et P2P), et l'identification des applications, dans laquelle l'identification des applications des utilisateurs finaux (par exemple, BitTorrent et Skype) est souhaitée. En 2018, wang and al. [59] ont proposé un nouvel IDS appelé le système de détection d'intrusion hiérarchique spatio-temporel basé sur des caractéristiques (HAST-IDS). Ils ont appliqué CNNs et LSTM pour la classification du datasets ISCX2012. Lopez-Martin and al. [60] ont proposé une nouvelle technique pour NTC basée sur une combinaison de deux modèles d'apprentissage profond : CNN et LSTM qui peuvent être utilisés pour IoT traffic. Ils ont appliqué l'entête plus la charge utile comme caractéristique.

Kaiming He [61] a proposé un modèle de classification sur des caractéristiques des images. Il a appliqué le ResNet101 et ResNet200 pour la classification De ses datasets appelé CIFAR-10 ImageNet. Saining Xie [62] a proposé un modèle de classification sur des caractéristiques des images aussi mais Il a appliqué le ResNeXT pour la classification De ses datasets appelé ImageNet-5K ImageNet. Baohua Sun [63] a proposé un modèle de classification sur des caractéristiques des images mais il a modifié le nombre de couche ResNet. Il a utilisé nombre de couche égale à 400. La classification est faite avec une dataset appelé ImageNet.

**Tableau 2.2** : Travaux liés à la classification deep learning.

Recherche	Année	Datasets	Caractéristiques	Classificateur	Performances
<b>Mohammad lotfollahi [15]</b>	4 juin 2018	ISCX VPN non VPN	Header	Autoencoder et 1D-CNN	98%
<b>Wang-2018 [59]</b>	2018	ISCX2012	Header+payload	CNN+LSTM	98%
<b>Lopez-Martin [60]</b>	2017	Data from RedIRIS	Header+time Series	CNN+LSTM	90%
<b>Kaiming He [63]</b>	10 Dec 2015	CIFAR-10 ImageNet.	Les caractéristiques d'image	ResNet	96%
<b>Kaiming He [61]</b>	25 Jul 2016	CIFAR-10 ImageNet.	Les caractéristiques d'image	ResNet 1001-layer ResNet 200-layer	95%
<b>Saining Xie1 [62]</b>	11 Apr 2017	ImageNet-5K ImageNet.	Les caractéristiques d'image	ResNeXt	96%
<b>Baohua Sun [35]</b>	4 Juin 2019	ImageNet	Les caractéristiques d'image	ResNet 400- layer	82%

Ces travaux examinent un grand nombre d'études basées sur l'apprentissage machine et l'apprentissage profond. Comme le montre le **tableau 2.1**. Dans ces études, ils ont utilisés des algorithmes classiques de machine Learning tout en variant les Datasets. La majorité d'eux ont obtenue presque les mêmes performances (entre **94%** et **98%**). Par contre Drapergil [57] a atteint une performance égale à 80%. La cause de déséquilibre est probablement dans les domaines suivants (i) les ensembles de données de référence sont peu nombreux, bien que le même ensemble de données soit utilisé, et les méthodes d'extraction des échantillons utilisées par chaque institut varient. (ii) les paramètres d'évaluation ne sont pas uniformes, de nombreuses études n'évaluent que la précision du test, et le résultat est unilatéral.

Le **tableau 2.2**, Dans ces études, ils ont utilisées des algorithmes classiques de deep Learning tout en variant les Datasets. Ils ont obtenue des performances équilibrées (entre **90%** et**98%**). Par contre Baohua Sun [35] a atteint une performance égale à **82%**. La cause de déséquilibre est probablement dans le nombre de couches utilisées pour chaque algorithme. Nous concluons que chaque classificateur est adapté avec un Dataset bien précis, il suffit plusieurs tests pour atteindre des meilleures performances.

## 7 Conclusion

Dans ce chapitre, nous avons passé en revue la classification du trafic réseau. Premièrement nous avons expliquées la notion de trafic réseau ainsi que le flux de donnée et un paquet internet. Ensuite nous avons traitées c'est quoi un trafic vpn. De plus, nous avons cités le principe de classification de trafic réseau et on finit par les travaux connexes similaires de ce travail.

## **PARTIE II : EXPERIMENTATON**

Dans cette partie, nous décrivons la méthodologie que nous avons suivie pour évaluer et comparer les modèles de la classification du trafic réseau. Elle est composée de deux chapitres. Dans le premier chapitre, nous présentons la conception et la méthodologie de notre travail, tandis que dans le second chapitre, nous discutons des résultats de nos expériences.

# Chapitre 3 : Méthodologie

## 1 Introduction

Ce chapitre présente la méthodologie et les méthodes quand on a utilisé et aussi les outils de programmation, ainsi l'utilisation des algorithmes de classification afin que ces derniers nous aident à prédire si le trafic réseau est un trafic vpn /non-vpn, Tout d'abord, dans la section 2, nous présentons notre DataSet suivi par la section 3, qui parle sur les outils de la réalisation. Ensuite, dans la section 4, nous avons fait une description de système. Ensuite, dans la section 3.5, nous expliquons le prétraitement. En outre, dans la section 6, nous expliquons les méthodes utilisées pour la sélection d'attributs. Enfin, dans les sections 7, 8 et 9 respectivement, nous expliquons le découpage de notre ensemble de données en ensembles de formation et de test, l'apprentissage des données par des Algorithmes d'apprentissage machine et profond utilisés dans nos expériences et l'implémentation de notre modèle et aussi les métriques utilisés pour l'évaluation des performances.

## 2 DataSet

En raison que SONATRACH a un principe de sécurité très haut, l'obtention de ses propres données reste une demande illégale. Pour cela nous avons utilisé un DataSet publique adaptable à notre travail.

L'ensemble de données utilisé pour cette tâche est ISCXVPN collectés par l'Université de New Brunswick (UNB) en 2016.

Les modèles de classification du trafic réseau ne peuvent être construits que s'il existe un ensemble de données efficace. Un ensemble de données contenant une quantité appréciable de données de qualité imitant le temps réel ne peut que permettre de former et de tester un modèle de classification du trafic. Dans ce projet, le jeu de données ISCX est analysé et utilisé pour étudier l'efficacité de notre modèle de classification du trafic réseau.

Le jeu de données UNB ISCX Network Traffic (VPN-nonVPN) se compose de trafic réseau étiqueté, y compris un paquet complet au format pcap (flux générés par ISCXFlowMeter) sont également accessibles au public pour les chercheurs [64].

Le jeu de données ISCX étiqueté comporte 12 classes (**Tableau 3.1**), dont 6 classes de trafic régulier chiffré et 6 classes de protocole de trafic encapsulé. Le tableau suivant montre le contenu détaillé de DataSet.

Traffic Type	Content
Email	Email, Gmail ( SMTP, POP3,IMAP )
VPN-Email	
Chat	ICQ, AIM, Skype, Facebook, Hangouts
VPN-Chat	
Streaming	Vimeo, Youtube, Netflix, Spotify
VPN-Streaming	
File transfer	Skype, FTPS, SFTP
VPN-File transfer	
VoIP	Facebook, Skype, Hangouts, Voipbuster
VPN-VoIP	
P2P	uTorrent, Bittorrent
VPN-P2P	

**Tableau 3.1** : Contenu de l'ensemble de données de trafic réseau UNC ISCX [64].

Pour générer un ensemble de données représentatif du trafic réel dans ISCX, ils ont défini un ensemble de tâches, garantissant que leur ensemble de données est suffisamment riche en diversité et en quantité. ils ont créé des comptes pour les utilisateurs Alice et Bob afin d'utiliser des services tels que Skype, Facebook, etc.

ils ont capturé une session régulière et une session sur VPN, nous avons donc un total de 12 catégories de trafic.

**Tableau 3.2:** Nombre d'occurrence de classes.

Classe	Occurrence
Email	32566
VPN-Email	2767
Chat	404
VPN-Chat	1168
Streaming	51932
VPN-Streaming	115939
File Transfer	780
VPN-File Transfer	102569
VoIP	87334

<b>VPN-VoIP</b>	15094
<b>P2P</b>	108541
<b>VPN-P2P</b>	422698
<b>Total</b>	160153

### 3 Outil de la réalisation

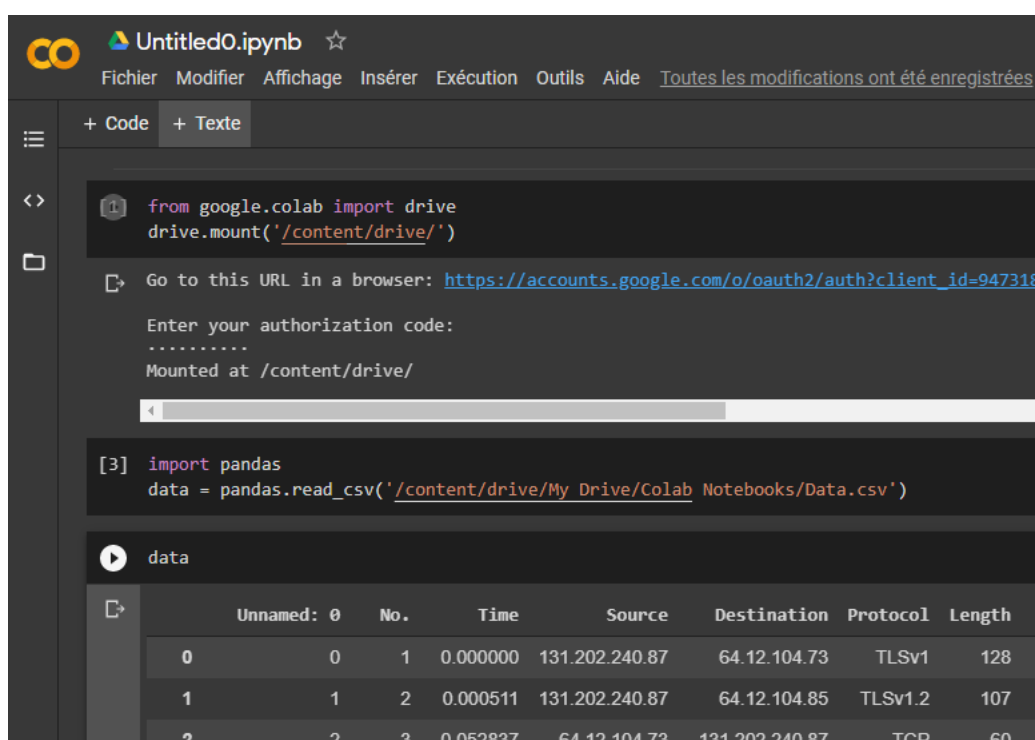
Dans cette partie nous allons présenter les principaux outils utilisés pour la mise en place de notre projet.

- ✚ Un ordinateur Toshiba Intel® Pentium® CPU N 3700 ayant 4Go de Ram est utilisé pour la
- ✚ Python 3.7 est utilisé comme langage de programmation qui est un langage open source orienté objet.
- ✚ Le jeu de données UNB ISCXVPN 2016.est sélectionné.
- ✚ Pandas (version utilisée 1.0.5): une bibliothèque utile pour l'analyse des données afin de présenter nos fonctionnalités [65].
- ✚ Mlxtend 0.17.2 est utilisé pour la sélection d'attributs qui est une bibliothèque python d'outils utiles pour les tâches quotidiennes de la science des données [66].
- ✚ Scikit learn (version utilisée 0.21.3): une bibliothèque d'apprentissage machine, elle expose une grande variété d'algorithmes d'apprentissage automatique, c'est le moteur de beaucoup d'application de l'intelligence artificielle et de science des données [67].
- ✚ Numpy (version utilisée 1.18.5) : extention de python, elle permet de créer directement un tableau depuis un fichier ou au contraire de sauvegarder un tableau dans un fichier, et manipuler des vecteurs, matrices et polynomes [68].
- ✚ Matplotlib (version utilisée 3.2.1): est une bibliothèque de traçage de graphique python 2D capable de générer des histogrammes, des spectres de puissance, des graphique a barre, différents diagrammes avec seulement quelques ligne de code [69].
- ✚ Knn, naïve bayes, svm, forêt aléatoire, arbre de décision et Logistic Reggresion sont utilisés comme des classificateurs dans notre projet.



- ✚ Tonsenflow et Keras (version utilisée pour tonsenflow 2.2.0 et keras 2.4.2) : des bibliothèques pratiques pour construire n'importe quel algorithme d'apprentissage profond [70].
- ✚ torch (version utilisée 1.5.1) est utilisée pour la classification des images.

Nous avons formé nos classificateurs en utilisant Google Colaboratory qui est un service gratuit de Cloud (**Figure 3.1**). Il consiste en des ordinateurs portables Python exécutables stockés dans Google Drive et connectés à un runtime basé sur le Cloud pour exécuter le code Python sur les GPU et TPU Nvidia Tesla K80.



**Figure 3.1** : Capture d'écran de la plateforme en ligne Google colab.

## 4 Description du système

Notre objectif est de construire différents systèmes de classification capables d'indiquer si le trafic est VPN ou non. Chaque système utilise une technique spécifique de sélection d'attributs ainsi que différents paradigmes de classification (**Figure 3.2**).

Tout d'abord, nous avons examiné 2 techniques de sélection d'attributs (Wrapper et Filter) et 6 paradigmes de classification (KNN, SVM, RF, Arbre de décision, LogisticRegression et Naive Bays), tout en faisant varier leurs paramètres. Nous avons

utilisé les mesures F-score et Accuracy et le Rappel pour évaluer nos systèmes et avons étayé notre analyse et notre discussion par de nombreux tests statistiques.

Dans la phase d'apprentissage profond nous avons utilisés la méthode « Two-dimensional embedding » dans le cadre de l'apprentissage transféré qui permet de convertir notre ensemble de données tabulaire en image pour qu'on puisse les classifiés par des paradigmes d'apprentissage profond (ResNet18).

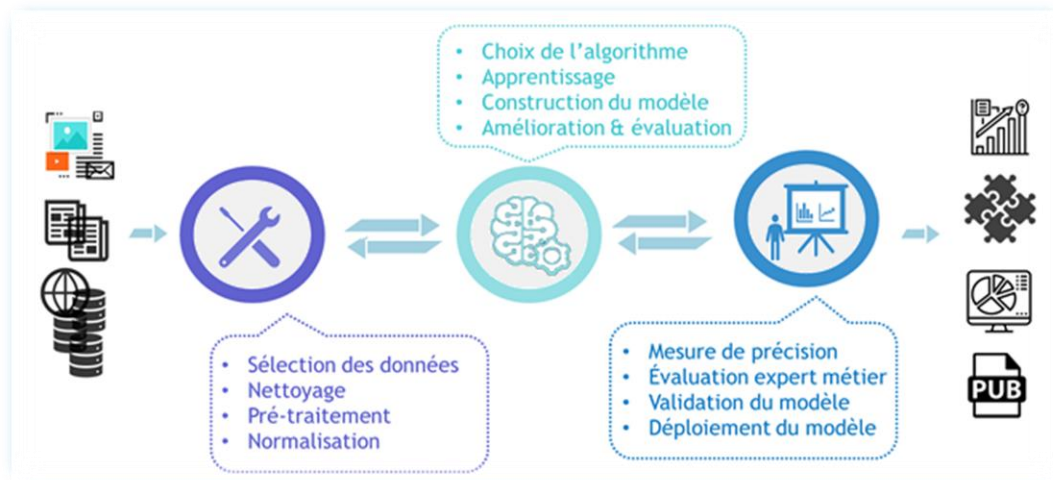


Figure 3.2 : mécanisme du système.

## 5 Preprocessing

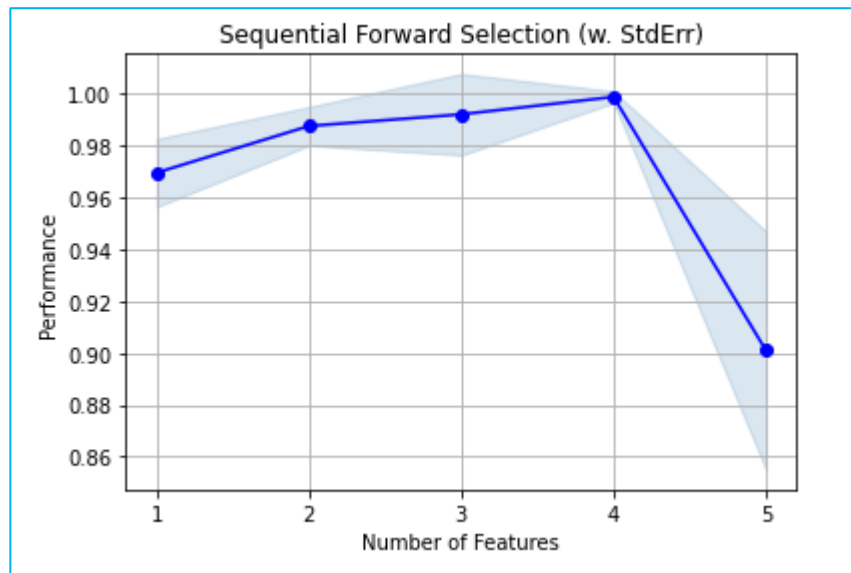
Les données utilisées pour ce travail sont de types catégoriels. Il contient des attributs de type entier, réel et texte. Pour pouvoir les classifiés par des paradigmes de classification, il est obligatoire de passer par la phase de "preprocessing". Cette dernière définis par les étapes suivantes :

- ✚ La transformation de l'ensemble de données qui contient plusieurs fichiers de format PCAP en CSV et les fusionnées en un seul fichier.
- ✚ L'étiquetage des données en 12 classes.
- ✚ L'utilisation de la méthode Integer encoded pour l'obtention d'un tableau numérique.
- ✚ La suppression des valeurs manquantes et aberrantes.

Ce processus a présenté plusieurs étapes à prendre en compte lors du prétraitement des données brutes de trafic réseau pour l'exploration de données (par exemple, faire des prévisions), Certaines de ces étapes sont essentielles, d'autres peuvent être facultatives, A fourni une étude de cas en utilisant un ensemble de données librement disponibles. Les résultats montrent que les étapes sont effectivement essentielles pour obtenir des résultats.

## 6 La sélection d'attributs

Afin d'augmenter les performances de la classification par la sélection des meilleurs attributs et en conséquence réduire la dimensionnalité de la matrice, nous avons utilisé les deux méthodes intitulées Wrapper et Filter, cette méthode calcul la relation entre les attributs et les classes.



**Figure 3.3** : le taux d'importance des attributs sélectionnés.

## 7 Cross-validation

Pour l'évaluation du modèle, les méthodes Hold-out et K-fold cross validation sont utilisées. Pour le hold out, le dataset est subdivisé en deux parties, 33% pour l'évaluation et 67% pour l'apprentissage. Pour la deuxième méthode, le K=10. Lorsque tous les modèles sont formés, un score de performance global est calculé en prenant la moyenne des résultats sur les 10 splits. Nous avons mélangé nos données pour générer différentes combinaisons afin de nous assurer que les événements sont présents dans l'ensemble de test.

```
scores = cross_val_score(RandomForestClassifier(max_depth=3, random_state=3, n_estimators = 400), X, y, cv=5)
print("Accuracy", scores.mean())
```

**Figure 3.4** : configuration 10-fold cross validation.

```
#hold out
import numpy as np
from sklearn.model_selection import train_test_split
X = data.drop('class', axis= 1)
y = data['class']
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.33, random_state=0)
```

Figure 3.5 : configuration Hold out.

## 8 Les algorithmes de classification

Pour comparer les performances de chaque système, nous avons examiné 6 classificateurs d'apprentissage machine supervisé, comme décrit dans le tableau. Nous avons implémenté les 6 premiers classificateurs en Python en utilisant la bibliothèque Scikit learn. Il est intéressant de souligner que nous avons effectué une mise à l'échelle des caractéristiques de nos vecteurs de caractéristiques en utilisant le correcteur standard de prétraitement Scikit learn avant de former nos modèles afin de s'assurer que nos caractéristiques prennent des valeurs similaires.

Tableau 3.3 : Configuration de l'apprentissage automatique.

Classifieur	Paramètres	Valeurs
k-nearest neighbor (KNN_1)	n_neighbors	1
k-nearest neighbor (KNN_2)	n_neighbors	11
Support vector machine (SVM_1)	kernel	Linear
Support vector machine (SVM_2)	kernel	Poly
Support vector machine (SVM_3)	kernel	Rbf
Gaussian Naive bays (NB_1)	\	\
Bernoulli Naive bays (NB_2)	\	\
Arbre de décision (AR)	criterion max_depth	entropy 3
Random Forest (RF_1)	max_depth n_estimators	2 100
Random Forest (RF_2)	max_depth n_estimators	2 400
LogisticRegression	solver	Lbfgs

(LR)

D'autre part, on a utilisé l'architecture ResNet et précisément la configuration qui utilise 18 couches (ResNet18), c'est une architecture optimisé de type CNN. Cette méthode permet de classifier notre ensemble de données que nous le convertir en image avec la méthode « two-dimensional embedding ».

## 9 Métriques utilisés

✚ **Précision :**

$$précision = \frac{TP}{TP + FP} \quad (12)$$

✚ **Rappel :**

$$rappel = \frac{TP}{TP + FN} \quad (13)$$

✚ **F\_mesure :**

$$f\_mesure = \frac{2 * précision * rappel}{précision + rappel} \quad (14)$$

✚ **Classification report :**

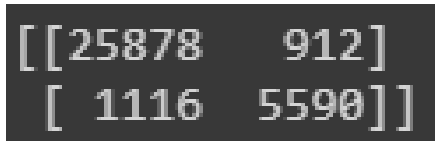
`sklearn.metrics.classification_report` : résumé textuel de la précision, rappel, score F1 pour chaque classe.

	precision	recall	f1-score	support
0.0	0.96	0.97	0.96	26798
1.0	0.86	0.83	0.85	6706
accuracy			0.94	33496
macro avg	0.91	0.90	0.90	33496
weighted avg	0.94	0.94	0.94	33496

Figure 3.6 : configuration d'une classification report.

**✚ Matrice de confusion :**

Une Confusion Matrix est un résumé des résultats de prédictions sur un problème de classification. Les prédictions correctes et incorrectes sont mises en lumière et réparties par classe. Les résultats sont ainsi comparés avec les valeurs réelles [71].



```
[[25878  912]
 [ 1116 5590]]
```

**Figure 3.7 :** exemple d'une matrice de confusion.

**True Negative (TN):** 25878.

**False Négative (FN) :** 912.

**Faux Positif (FP) :** 1116.

**True Positive (TP) :** 5590.

## 10 Conclusion

Ce chapitre présente le dispositif expérimental défini pour effectuer nos comparaisons expérimentales. Nous avons présenté notre ensemble de données et la façon dont nous l'avons divisé en ensemble de formation et d'essai afin de réaliser l'évaluation de notre système. Nous avons également présenté les outils que nous avons utilisés pour mener notre expérience et la topologie de notre système. Nous avons souligné le fait que notre étude expérimentale repose sur différents ensembles de caractéristiques spectrales et de modèles d'apprentissage machine et avons exposé leurs paramètres.

# Chapitre 4: Résultats Expérimentaux et discussion

## 1 Introduction

Ce chapitre traite des résultats que nous avons obtenus au cours de nos expériences. Dans les sections 4.2 et 4.3, nous analysons et discutons les résultats de la première et de la deuxième série de nos expériences. Dans la section 4.4, nous parlons de la durée de formation de nos études expérimentales.

## 2 Première expérience

Dans cette étape, nous avons fait une expérimentation **machine learning** sur notre ensemble de données de type **float** et **integer** sans faire passer par **la sélection d'attributs**. La répartition de l'ensemble a été faite par la méthode **Hold out** de cross validation.

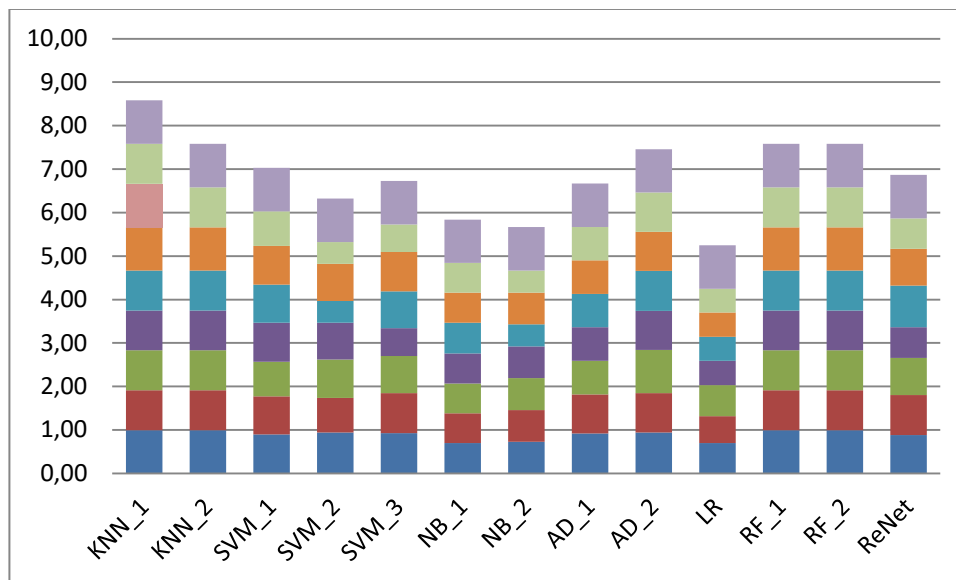
Les mesures de performances obtenues sont situées dans le **tableau 4.1** suivant :

**Tableau 4.1** : Résultats de classification machine learning sans sélections d'attributs avec Hold out.

Algorithme	Paramètre	f_mesure
<b>KNN_1</b>	n_neighbors = 40	0.92
<b>KNN_2</b>	n_neighbors = 1	0.92
<b>SVM_1</b>	Kernel = 'linear'	0.87
<b>SVM_2</b>	Kernel = 'poly'	0.80
<b>SVM_3</b>	Kernel = 'rbf'	0.92
<b>NB_1</b>	\	0.68
<b>NB_2</b>	\	0.73
<b>AD_1</b>	Criterion = 'entropy' Max_depth = 1	0.90
<b>AD_2</b>	Criterion = 'entropy' Max_depth = 3	0.91
<b>LR</b>	Solver = 'lbfgs'	0.62
<b>RF_1</b>	Max_depth = 1 n_estimators = 100	0.92

<b>RF_2</b>	Max_depth = 3	0.92
	n_estimators = 400	

Le graphe illustré à la **figure 4.1** représente la distribution des **f\_mesures** sur l'ensemble des événements pour chaque système. Nous observons que le **KNN** et **RF** présente les meilleures performances, car le **f\_mesure** est égale à **92%**. Les autres algorithmes tels que **SVM**, **AD** et **NB\_2** se situent entre 70 % et 90%. En outre, les performances du **LR** et **NB\_1** sont très faible par rapport (**69%** et **68%**).



**Figure 4.1** : Comparaison entre f\_mesures des algorithmes machine learning avec une répartition hold out.

### 3 Deuxième expérience

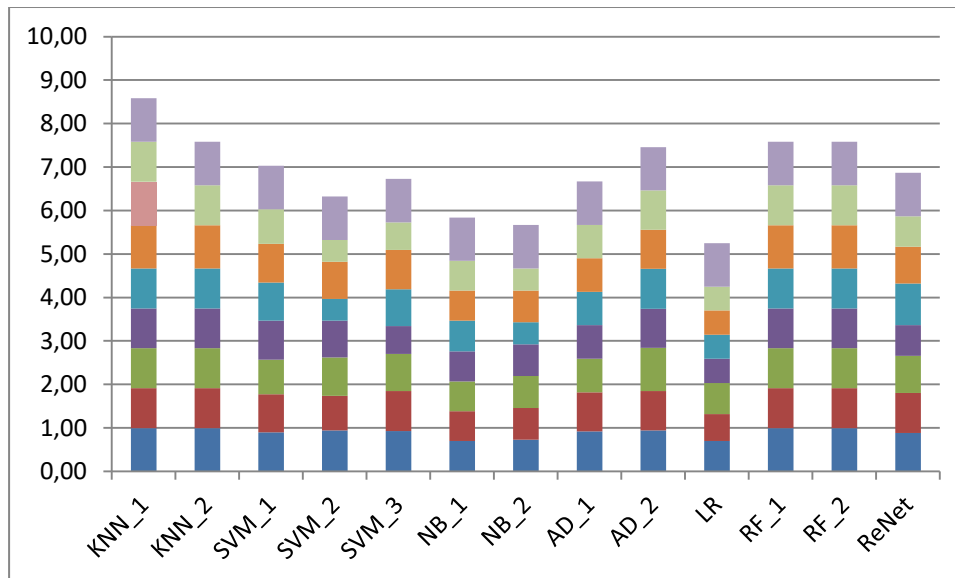
Dans cette étape, nous avons fait une expérimentation **machine learning** sur notre ensemble de données de type **float** et **integer** sans faire passer par **la sélection d'attributs**. La répartition de l'ensemble a été faite par la méthode **K\_fold** cross validation avec **K=10**. Les mesures de performances obtenues sont situées dans le **tableau 4.2** suivant :



**Tableau 4.2** : Résultats de classification machine learning sans sélections d'attributs avec 10\_fold cross validation.

Algorithme	Paramètre	Précision moyenne
<b>KNN_1</b>	n_neighbors = 40	0.92
<b>KNN_2</b>	n_neighbors = 1	0.92
<b>SVM_1</b>	Kernel = 'linear'	0.80
<b>SVM_2</b>	Kernel = 'poly'	0.88
<b>SVM_3</b>	Kernel = 'rbf'	0.85
<b>NB_1</b>	\	0.69
<b>NB_2</b>	\	0.73
<b>AD_1</b>	Criterion = 'entropy' Max_depth = 1	0.77
<b>AD_2</b>	Criterion = 'entropy' Max_depth = 3	0.90
<b>LR</b>	Solver = 'lbfgs'	0.71
<b>RF_1</b>	Max_depth = 1 n_estimators = 100	0.92
<b>RF_2</b>	Max_depth = 3 n_estimators = 400	0.92

Le graphe illustré à la **figure 4.2** représente la distribution des scores moyens de précision sur l'ensemble des événements pour chaque système. Nous observons que le **KNN** et **RF** présentent les meilleures performances avec une précision **92%**. Les autres algorithmes comme **NB\_2**, **LG**, **AD** et **SVM** donnent des précisions entre **70%** et **90%**. En outre, la précision **NB\_1** est très faible par rapport (**69%**).



**Figure 4.2 :** Comparaison entre f\_mesures des algorithmes machine learning avec une répartition 10\_fold.

#### 4 Troisième expérience

Dans cette étape, nous avons faire une expérimentation **machine learning** sur notre ensemble de données de type **float** et **integer** passant par **la sélection d'attributs**. La répartition de l'ensemble a été faite par la méthode **hold-out** cross validation.

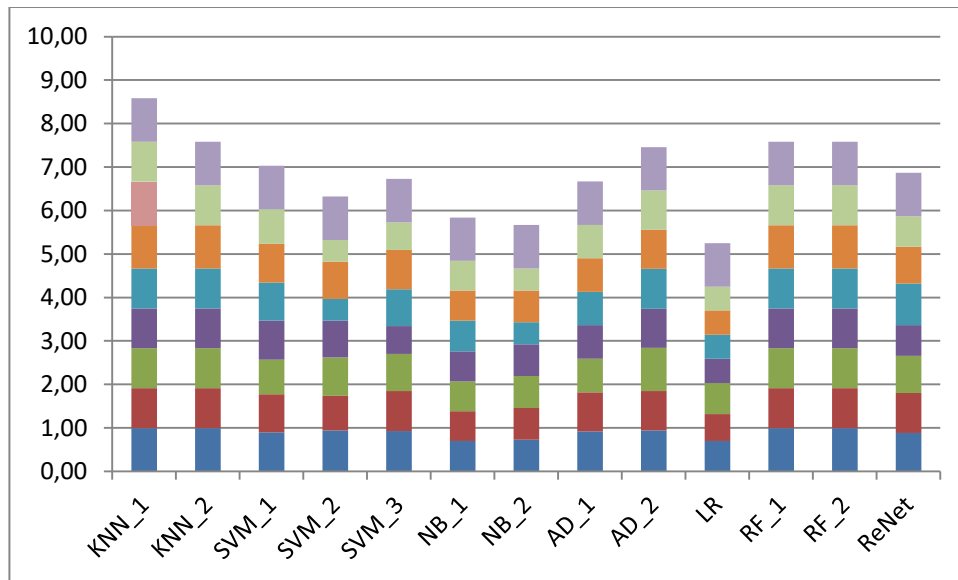
##### La méthode Filter :

Dans cette méthode, nous avons fixé le **threshold = 0.4**, cette dernière représente le seuil minimal de l'importance des attributs par rapport à la classe. Les mesures de performances obtenues sont situées dans le **tableau 4.3** suivant :

**Tableau 4.3** : Résultats de classification machine learning en utilisant ma méthode filter de sélections d'attributs avec Hold out.

Algorithme	Paramètre	f_mesure
<b>KNN_1</b>	n_neighbors = 40	0.92
<b>KNN_2</b>	n_neighbors = 1	0.92
<b>SVM_1</b>	Kernel = 'linear'	0.90
<b>SVM_2</b>	Kernel = 'poly'	0.85
<b>SVM_3</b>	Kernel = 'rbf'	0.64
<b>NB_1</b>	\	0.69
<b>NB_2</b>	\	0.73
<b>AD_1</b>	Criterion = 'entropy' Max_depth = 1	0.77
<b>AD_2</b>	Criterion = 'entropy' Max_depth = 3	0.90
<b>LR</b>	Solver = 'lbfgs'	0.56
<b>RF_1</b>	Max_depth = 1 n_estimators = 100	0.92
<b>RF_2</b>	Max_depth = 3 n_estimators = 400	0.92

Le graphe illustré à la **figure 4.3** représente la distribution des **f\_mesures** pour chaque algorithme en utilisant la **méthode filter** de sélection d'attributs. Nous observons que le **KNN**, et **RF** présente les meilleures performances, car le f\_mesure est égale à **92%**. Les autres algorithmes tels que **SVM\_1**, **SVM\_2**, **AD\_1**, **AD\_1**, **NB\_2** se situent entre **73%** et **90%**. En outre, les performances du **NB\_1**, **SVM\_3** et **LR** sont très faibles par rapport (**69%**, **64%** et **56%**).



**Figure 4.3:** Comparaison entre f\_mesures des algorithmes machine learning en utilisant hold out et la méthode filter.

**La méthode wrapper :**

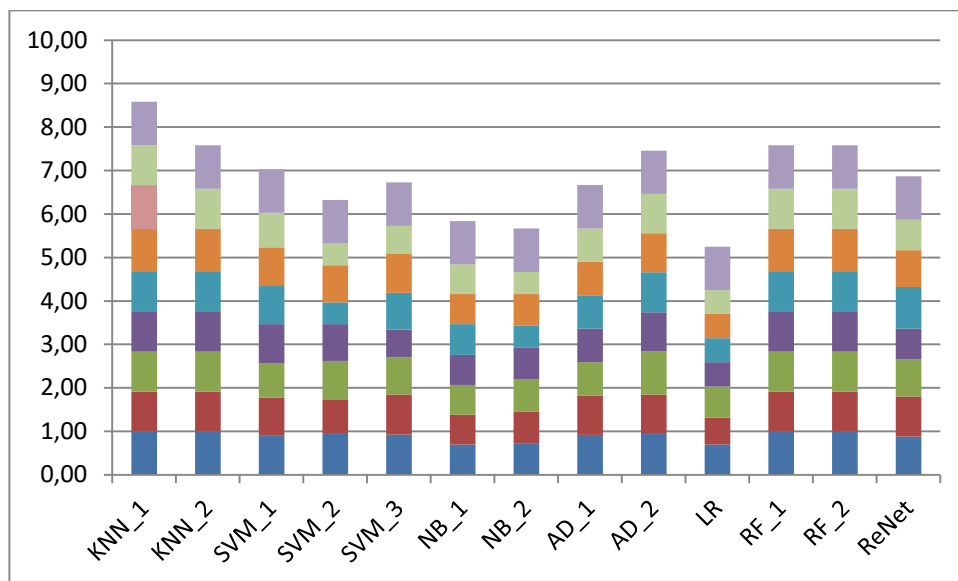
Les mesures de performances obtenues sont situées dans le **tableau4.4** suivant :

**Tableau 4.4 :** Résultats de classification machine learning en utilisant ma méthode wrapper de sélections d'attributs avec Hold out.

Algorithme	Paramètre	f_mesure
<b>KNN_1</b>	n_neighbors = 40	0,92
<b>KNN_2</b>	n_neighbors = 1	0.92
<b>SVM_1</b>	Kernel = 'linear'	0.87
<b>SVM_2</b>	Kernel = 'poly'	0.80
<b>SVM_3</b>	Kernel = 'rbf'	0.85
<b>NB_1</b>	\	0.71
<b>NB_2</b>	\	0.51
<b>AD_1</b>	Criterion = 'entropy' Max_depth = 1	0.77
<b>AD_2</b>	Criterion = 'entropy' Max_depth = 3	0.92
<b>LR</b>	Solver = 'lbfgs'	0.55

<b>RF_1</b>	Max_depth = 1 n_estimators = 100	0.92
<b>RF_2</b>	Max_depth = 3 n_estimators = 400	0.92

Le graphe illustré à la **figure 4.4** représente la distribution des **f\_mesures** pour chaque algorithme en utilisant la **méthode wrapper** de sélection d'attributs. Nous observons que le **RF** et **KNN** présente les meilleures performances, car le f\_mesure est égale à **92%**. Les autres algorithmes tels que **SVM**, **NB\_1**, **AD** se situent entre **70%** et **90%**. En outre, les performances du **NB\_2** et **LR** sont très faibles par rapport (**51%** et **55%**).



**Figure 4.4 :** Comparaison entre f\_mesures des algorithmes machine learning en utilisant hold out et la méthode wrapper.

## 5 Quatrième expérience

Dans cette étape, nous avons faire une expérimentation **machine learning** sur notre ensemble de données de type **float** et **integer** passant par la **sélection d'attributs**. La répartition de l'ensemble a été faite par la méthode **10\_fold cross validation** cross validation.

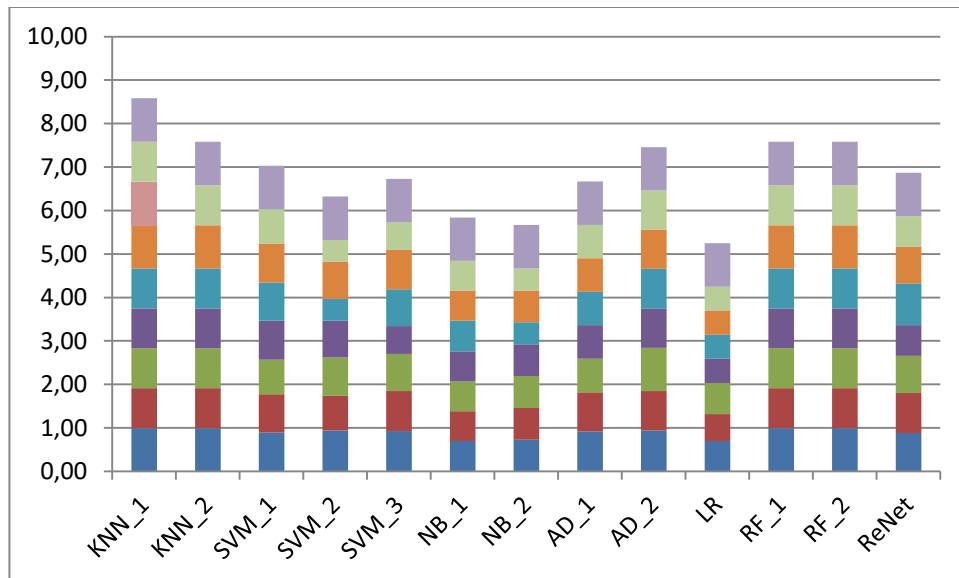
### La méthode filter

Dans cette méthode, nous avons fixé le **threshold = 0.6**. Les mesures de performances obtenues sont situées dans le **tableau 4.5** suivant :

**Tableau 4.5** : Résultats de classification machine learning en utilisant la méthode filter de sélections d'attributs avec K\_fold cross validation.

Algorithme	Paramètre	Précision moyenne
<b>KNN_1</b>	n_neighbors = 40	0.99
<b>KNN_2</b>	n_neighbors = 1	0.99
<b>SVM_1</b>	Kernel = 'linear'	0.89
<b>SVM_2</b>	Kernel = 'poly'	0.85
<b>SVM_3</b>	Kernel = 'rbf'	0.90
<b>NB_1</b>	\	0.69
<b>NB_2</b>	\	0.73
<b>AD_1</b>	Criterion = 'entropy' Max_depth = 1	0.77
<b>AD_2</b>	Criterion = 'entropy' Max_depth = 3	0.92
<b>LR</b>	Solver = 'lbfgs'	0.56
<b>RF_1</b>	Max_depth = 1 n_estimators = 100	0.99
<b>RF_2</b>	Max_depth = 3 n_estimators = 400	0.99

Le graphe illustré à la **figure 4.5** représente la distribution des **f\_mesures** pour chaque algorithme en utilisant la **méthode filter** de sélection d'attributs. Nous observons que le **KNN**, et **RF** présente les meilleures performances, car la précision est égale à **92%**. Les autres algorithmes tels que **SVM**, **NB\_2** et **AD** se situent entre **70%** et **90%**. En outre, les performances du **NB\_1** et **LR** sont très faibles par rapport (**69%** et **56%**).



**Figure 4.5 :** Comparaison entre f\_mesures des algorithmes machine learning en utilisant 10\_fold et la méthode filter.

**La méthode wrapper**

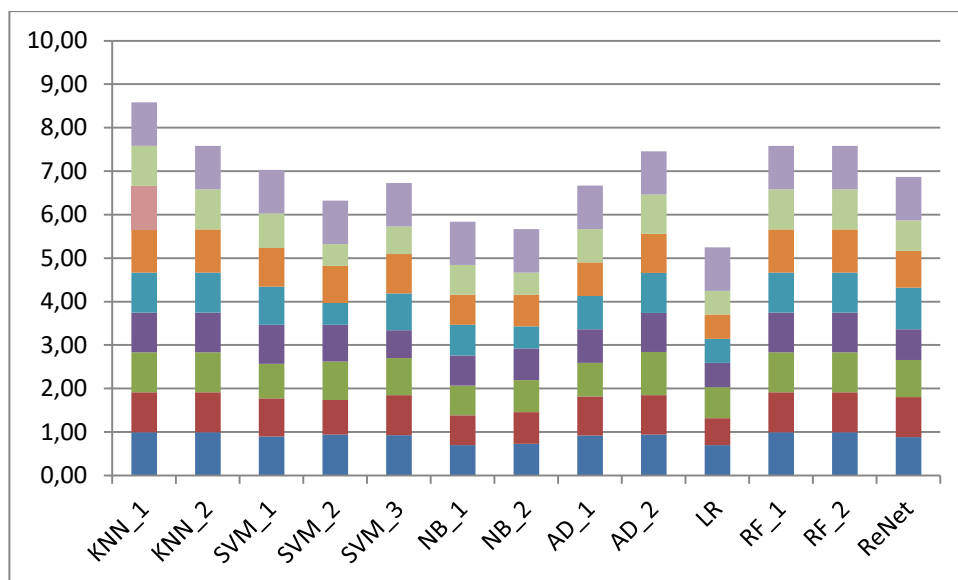
Les mesures de performances obtenues sont situées dans le **tableau 4.6** suivant :

**Tableau 4.6 :** Résultats de classification machine learning en utilisant ma méthode wrapper de sélections d’attributs avec 10\_fold cross validation.

Algorithme	Paramètre	Précision moyenne
<b>KNN_1</b>	n_neighbors = 40	0.99
<b>KNN_2</b>	n_neighbors = 1	0.99
<b>SVM_1</b>	Kernel = ‘linear’	0.90
<b>SVM_2</b>	Kernel = ‘poly’	0.94
<b>SVM_3</b>	Kernel = ‘rbf’	0.93
<b>NB_1</b>	\	0.70
<b>NB_2</b>	\	0.73
<b>AD_1</b>	Criterion = ‘entropy’ Max_depth = 1	0.92
<b>AD_2</b>	Criterion = ‘entropy’ Max_depth = 3	0.94
<b>LR</b>	Solver = ‘lbfgs’	0.70

<b>RF_1</b>	Max_depth = 1	0.99
	n_estimators = 100	
<b>RF_2</b>	Max_depth = 3	0.99
	n_estimators = 400	

Le graphe illustré à la **figure 4.6** représente la distribution de la précision pour chaque algorithme en utilisant la **méthode wrapper** de sélection d'attributs. Nous observons que le **KNN** et **RF** présente les meilleures performances, car la précision est égale à **99%**. Les autres algorithmes se situent entre **70%** et **94**.



**Figure 4.6 :** Comparaison entre f\_mesures des algorithmes machine learning en utilisant 10\_fold et la méthode wrapper.

## 6 Cinquième expérience

Dans cette étape, nous avons examinées une expérimentation de l'algorithme **ResNet18** sur notre ensemble de données de type **float** et **integer** passant par **la conversion en image** en projetant les valeurs de toutes les colonnes pour chaque ligne dans une image noire.

Pour savoir si la performance pouvait être affectée par **les positions des caractéristiques sur l'image**. Nous avons changés ses positions de gauche à droite et les autres conditions sont toutes les mêmes.



La répartition de l'ensemble a été faite par la méthode **hold out** cross validation en prenant **20%** de notre DataSet pour le test et le reste pour la formation.

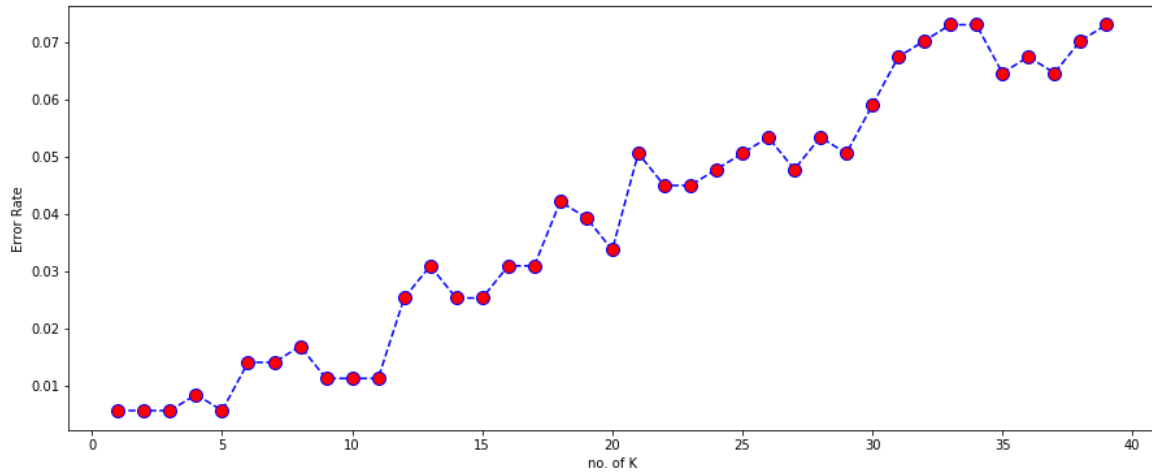
Les mesures de performances obtenues sont situées dans le **tableau 4.7** suivant :

**Tableau 4.7** : Résultats de classification transfer learning avec Hold out.

Algorithme	Epoques	Accuracy
<b>ResNet18_1</b>	EPOCH 1/ 3	0.88
	EPOCH 2/ 3	0.92
	EPOCH 3/ 3	0.86
<b>ResNet18_2</b>	EPOCH 1/ 3	0.70
	EPOCH 2/ 3	0.96
	EPOCH 3/ 3	0.85

Nous observons que le Resnet atteint une performance égale à **96%** avec la deuxième méthode de positionnement. Donc nous concluons que la deuxième méthode de positionnement est la meilleure pour nos DataSet.

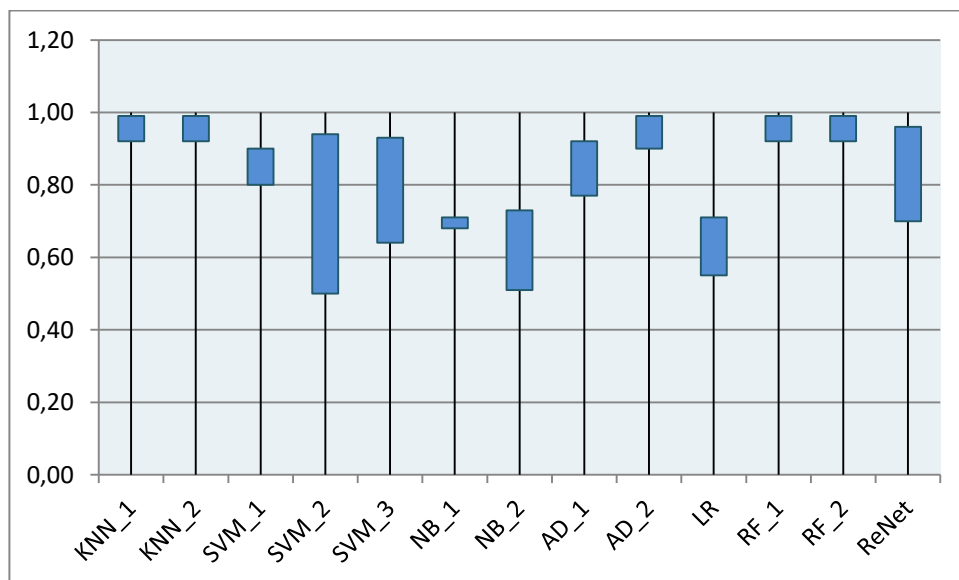
Pour générer un bon KNN, nous avons implémenté une fonction appelée **Error rate** qui donne le meilleur **K** pour notre classification, afin d’obtenir des meilleures performances. Le résultat est dans la **figure 4.7**.



**Figure 4.7 :** Error Rate.

Finalement nous avons fait une comparaison entre toutes les performances des algorithmes utilisés pour cette classification et nous avons trouvés que les algorithmes machine learning donnent des meilleures performances par rapport au l’algorithme **Resnet**.

Les meilleurs algorithmes sont le **random forest** et le **KneighborsClassifier** qui nous donne une précision égale à **99%** en utilisant la méthode **wrapper** de sélection d’attributs. Pour cela nous choisissons pour notre modèle le random forest.



**Figure 4.8 :** La comparaison entre tous les algorithmes utilisés.

## 7 Conclusion

Dans ce chapitre, nous avons fait des expérimentation sur plusieurs algorithmes de classification en situant KNN, SVM, Naive Bays, arbre de décision, foret aléatoire, logistic regression, et ResNet, et nous trouvons que les algorithmes de machine learning donnent des meilleurs performances par rapport à celui de transfer et deep learning.

Nous avons obtenus une précision égale à 0,99 quand nous utilisons l'algorithme Random forest, donc nous avons fixé notre modèle de classification qui classifie le trafic vpn par l'algorithme Random Forest.

# CONCLUSION

## 1 Contributions et résumé des résultats expérimentaux

Le travail, ainsi présenté dans ce mémoire est lié au domaine de la sécurité des réseaux plus précisément de la classification du trafic réseau chiffré.

Le but principal de ce mémoire était de mener une analyse empirique et des comparaisons entre les systèmes de classification du trafic réseau vpn et non-vpn. À cette fin, nous avons mené deux expériences sur la classification du trafic réseau avec l'utilisation de l'ensemble de données ISCX (VPN-nonVPN) Network Traffic2016.

Dans la première expérience, nous avons utilisés un ensemble de données tabulaire pour la classification. Nous avons examiné deux différents techniques de sélection d'attribut (Wrapper et filter) et 6 paradigmes de classification d'apprentissage automatique (SVM, KNN, Random Forest, Arbre de décision, Naive baise et Logistique Régression).

Dans la deuxième expérience, nous avons utilisés un ensemble de données de format image. L'obtention de ce dernier ce faite par la transformation de nos DataSet tabulaire en image. Le ResNet18 est utilisé pour cette classification tout en variant leurs paramètres.

Nous avons choisi le f-score, précision et le rappel comme mesure d'évaluation des performances et nous avons fondé notre analyse et notre discussion sur de nombreux tests statistiques. De cette étude expérimentale, nous pouvons tirer les conclusions suivantes :

Nous avons proposé un nouveau modèle de classification du trafic réseau chiffré à base de signature fonctionnant sur la base des algorithmes de classification supervisée de la machine Learning et deep Learning.

Nous sommes intéressés au foret aléatoire (RF) et KneighborsClassifier (KNN) avec l'utilisation de méthode de sélection d'attribut wrapper, où les expérimentations ont confirmé son succès déjà prouvé dans ce domaine.

D'autre part, nous avons proposé dans le domaine de deep learning, la transformation de DataSet tabulaire en image par la méthode two dimensionnel embedding et l'utilisation de l'architecteur ResNet18.

Les données de la classification du trafic réseau ISCX VPN-NONVPN restent toujours le meilleur corpus de données libellées, qui permet aux concepteurs de classifier, de construire, d'évaluer et de comparer les performances de leurs systèmes avec d'autres concepteurs qui ont utilisé le même ensemble de données.

## 2 Limites et travaux futurs

Ce mémoire a révélé plusieurs domaines intéressants à améliorer. En nous basant sur les résultats des expériences, nous avons conclu que la transformation des données en image peut affecter négativement la performance des systèmes de classification du trafic réseau en deep Learning car les données tabulaires qui se chevauchent dans la classification du trafic réseau ne peuvent pas être bien représentées par des images.

Au cours de ce travail, nous avons rencontré plusieurs difficultés concernant la conversion des Datas de type catégorielle en format adaptable aux algorithmes de classification (pour le machine Learning : format numérique et deep Learning : format image).

De plus, le processus de nos PCs portables ne supporte pas une très grand quantité de données. De plus, même avec le mode d'exécution TPU Nvidia Tesla K80, le temps de formation est encore énorme.

Pour les travaux future, on aimerait bien faire une classification en temps réel des données chiffrés et réaliser une plateforme qui classifie et calcule le taux de trafic chiffrés afin d'aider l'entreprise à gérer le trafic réseau.

Ce domaine de recherche est intéressant car il contribue directement au développement de la tarification intelligente. Nous avons acquis des connaissances et de nombreuses compétences au cours des 8 derniers mois, telles que : les bases de l'apprentissage automatique ainsi que l'apprentissage profond et les étapes clés pour mener des expériences d'apprentissage automatique et profond appropriées. Nous avons également appris à analyser les résultats des expériences sur la base de tests statistiques. De plus, nous avons maîtrisé Python et découvert la plateforme collaborative de Google que nous continuerons à utiliser pour de futurs projets d'apprentissage automatique et profond.

## REFERENCES

- [1] Ma, J., Levchenko, K., Kreibich, C., Savage, S., et Voelker, G. M. (2006). « *Unexpected means of protocol inference* ». In the Proceedings of the 6th ACM SIGCOMM conference on Internet measurement.
- [2] P. Velan, M. Cermak, P. Celeda and M. Drasar, "A survey of methods for encrypted traffic classification and analysis", International Journal of Network Management, vol. 25, no. 5, pp. 355-374, 2015.
- [3] e-marketing. Machine Learning. [https://www.e-marketing.fr/Definitions\\_Glossaire/Machine-learning-305604.htm](https://www.e-marketing.fr/Definitions_Glossaire/Machine-learning-305604.htm), consulté en Mars 2020.
- [4] Benmammar. (2017). « machine learning ». Université de telemcen.
- [5] Nathalie Toulon. (Novembre 2018). « deep learning et agriculture : comprendre le potentiel et les défis à relever. <https://innovin.fr/index.php/fr/medias/toutes-les-actus/45-files/851-deep-learning-et-agriculture-comprendre-le-potentiel-et-les-defis-a-relever>. consulté en Mars 2020.
- [6] Educba : Machine learning models.<https://www.educba.com/machine-learning-models/#> . consulter en Mars 2020.
- [7] Mohammad Waseem.19 nov 2019. « How To Perform Logistic Regression In Python ». <https://www.edureka.co/blog/logistic-regression-in-python/>. Consulté en mars 2020.
- [8] Usman.M. (2020). « Implementing SVM and Kernel SVM with Python's Scikit-Learn ». <https://stackabuse.com/implementing-svm-and-kernel-svm-with-pythons-scikit-learn/>. Consulter en Mars 2020.
- [9] Sklearn-learn developers (BSD license). 2007-2019. « decision trees ». <https://scikit-learn.org/stable/modules/tree.html>. Consulté en Mars 2020.
- [10] Leo Breiman. (january,2001). « *Random forest* ».University of California.
- [11] Olivier François. 2005. « *Comment Améliorer le Classifieur de Bayes Naïf ?* ».
- [12] Miloud Aouidate Amel. 30 avril 2014. « *Réduction à l'aide d'une approche basée sur les KNNs des alertes fausses positives issues d'un détecteur d'intrusions réseau* ». Université des sciences et de technologie houari boumedién USTHB.
- [13] Jason Brownlee.(December 20, 2017) . « *Deep Learning for Computer Vision* ». A Gentle Introduction to Transfer Learning for Deep Learning

- [14] Angelina Fausto. (19 septembre, 2018). « *Les Méthodes de transfert learning* ».
- [15] Mohamed,L. ,Mahdi,J.S. ,ramin,S.H. ,Mohammdsadegh,S. (4 jul 2018 ). « *Deep Packet : A novel approach for encrypted traffic classification using Deep learning* ». Sharif university of technology, tahrán, Iran.
- [16] Shahbaz,R .,Xin,L .(2019). « *Deep learning for encrypted traffic classification :An Overview* ».IEEE communication magazine .
- [17] Sehgal, A., & Kehtarnavaz, N. (2018). « *A convolutional neural network smartphone app for real-time voice activity detection* ». IEEE Access, 6, 9017-9026.
- [18] Vemula, H. C. (2018). « *Multiple Drone Detection and Acoustic Scene Classification with Deep Learning* ».Wright State University.
- [19] Sarthak Vajpayee. (Sep 7, 2019). « *Towards data science : AI-powered indian license plate detector* ».
- [20] Udeme Udofino. (Feb 13, 2018). « *Basic Overview of Convolutional neural Network (CNN)* ».
- [21] PyTorch ResNet. « *Building Residual Network son PyTorch* ».  
<https://missinglink.ai/guides/pytorch/pytorch-resnet-building-training-scaling-residual-networks-pytorch/> . Consulté en Mai 2020.
- [22] Mr. Mint. « *Machine Learning made easy* ».<https://mrmint.fr/datapreprocessing-feature-scaling-python> . Consulté en Mai 2020.
- [23] Microsoft azure. « *Machine Learning* ».<https://docs.microsoft.com/frfr/azure/machine-learning/team-data-science-process/prepare-data> . Consulté en Mai 2020.
- [24] Openclassroom.Initiez vous au machine learning.  
<https://openclassrooms.com/fr/courses/4011851-initiez-vous-au-machine-learning/4020631-exploitez-votre-jeu-de-donnees>. Consulté en Mai 2020
- [25] Jason Brownlee. July 28, 2017. « *why one-hot encode data in machine learning* ».
- [26] George Novack.7 jun 2020. « *N machine learning Building a One Hot Encoding Layer with TensorFlow* ».
- [27] Alakh Sethi. March 6, 2020. « *One-hot encoding vs label encoding using scikit learn* ».
- [28] LAROUMAGNE, F. Le machine Learning automatisé.  
<https://jedha.co/blog/2018/10/17/le-machine-learning-automatise/>. Consulté en Mai 2020.
- [29] Cambridge Spark. Sep 3,2019. « *Tutorial : Introduction to missing data imputation.* »
- [30] Jim Frost. « *5 ways to find outliers in your data.* ».
- [31] Jason Brownlee. November 27, 2019. « *Data Preparation How to Choose a Feature Selection Method For Machine Learning* ».

- [32] R. Kohavi and G. John, « *Wrapper for Feature Subset Selection* », *Artificial Intelligence* 97 (1-2) (1997) 273- 324.
- [33] P. Pudil and J. Novovicova, « *Novel methods for subset selection with respect to problem knowledge* », *IEEE Intell. Syst.* 13 (2) (1998) 66–74.
- [34] Y. Sun, S. Todorovic, and S. Goodison, « *Local- Learning-Based Feature Selection for High- Dimensional Data Analysis* », *TPAMI* 32 (9) (2010) 1610–1626.
- [35] Baohua Sun, Lin Yang, Wenhan Zhang, Michael Lin, Patrick Dong, Charles Young, Jason Dong. (4 Juin, 2019). « *SuperTML: Two-Dimensional Word Embedding for the Precognition on Structured Tabular Data* ».
- [36] Openclassroom. évaluez et améliorez les performances d'un modèle de machine learning .<https://openclassrooms.com/fr/courses/4297211-evaluez-et-ameliorer-les-performances-dun-modele-de-machine-learning> . Consulté en avril 2020.
- [37] Mr. Mint. Machine Learning made easy . <https://mrmint.fr/aborder-unprobleme-de-machine-learning-partie-2> . consulté en avril 2020.
- [38] Medium : hold-out vs. Cross-validation in Machine Learning . <https://medium.com/@ejaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f> . Consulté en Avril 2020.
- [39] Xavier Dupre. Bien démarrer un projet de machine learning. <https://studylibfr.com/doc/2015890/classification-et-apprentissage-actif-%C3%A0-partir-d-un-flux->. Consulté en Mai 2020.
- [40] R,Zhong . 2011. « *Dynamic assignment, surveillance and control for traffic network with uncertainties* »
- [41] LeMagIT : définition de paquet. <https://www.lemagit.fr/definition/Paquet>. Consulté en Mars 2020.
- [42] Steve Petryschuk. March 19 2019. « *NetFlow Basics: An Introduction to Monitoring Network Traffic* ».
- [43] Schneider, P. (1996). « *TCP/IP traffic Classification Based on port numbers* ». Division OfApplied Sciences, Cambridge, MA, 2138.
- [44] Moore, A. W., et Papagiannaki, K. (2005). « *Toward the accurate identification of network applications* ». In the International Workshop on Passive and Active Network Measurement.
- [45] Sen, S., Spatscheck, O., et Wang, D. (2004). « *Accurate, scalable in-network identification of p2p traffic using application signatures* ». In the Proceedings of the 13th international conference on World Wide Web.



- [46] Paxson, V. (1999). « *Bro: a system for detecting network intruders in real-time* ». *Computer networks*, 31(23), 2435-2463.
- [47] Finamore, A., Mellia, M., Meo, M., et Rossi, D. (2010). « *Kiss: Stochastic packet inspection classifier for udp traffic* ». *IEEE/ACM Transactions on Networking*, 18(5), 1505-1515.
- [48] Khakpour, A. R., et Liu, A. X. (2009). « *High-speed flow nature identification* ». In the *Distributed Computing Systems, 2009. ICDCS'09. 29th IEEE International Conference on*.
- [49] Karagiannis, T., Papagiannaki, K., et Faloutsos, M. (2005). « *BLINC: multilevel traffic classification in the dark* ». In the *ACM SIGCOMM Computer Communication Review*.
- [50] Iliofotou, M., Pappu, P., Faloutsos, M., Mitzenmacher, M., Singh, S., et Varghese, G. (2007). « *Network monitoring using traffic dispersion graphs (tdgs)* ». In the *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*.
- [51] Iliofotou, M., Kim, H.-c., Faloutsos, M., Mitzenmacher, M., Pappu, P., et Varghese, G. (2009). « *Graph-based p2p traffic classification at the internet backbone* ». In the *INFOCOM Workshops 2009, IEEE*.
- [52] Nguyen, T. T., et Armitage, G. (2008). « *A survey of techniques for internet traffic classification using machine learning* ». *IEEE Communications Surveys & Tutorials*, 10(4), 56-76.
- [53] VPNoverview.com VPN explained: How does it work? Why would you use it ?. <https://vpnoverview.com/vpn%20information/what-is-a-vpn/>. Consulté en Mars 2020.
- [54] Wei wang , Ming zhu and al.(2017). « *End-to-end encrypted traffic classification with one-dimensional convolution neural network* ». *University of science and technology f china*.
- [55] Nguyen, T. T., et Armitage, G. (2006). « *Training on multiple sub-flows to optimise the use of machine learning classifiers in real-world ip networks* ». In the *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*.
- [56] Claffy, K. C., Braun, H.-W., et Polyzos, G. C. (1995). « *A parameterizable methodology for Internet traffic flow profiling* ». *IEEE Journal on selected areas in communications*, 13(8), 1481-1494.
- [57] SainingX.Ross, Girshick2.Piotr.Dollar, 2.Zhuowen, Tu1.Kaiming, He2.(11Apr, 2017). « *Aggregated Residual Transformations for Deep Neural Networks* ».
- [58] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun and A. A. Ghorbani, « *Characterization of encrypted and VPN traffic using time-related features* », In *Proceedings of the 2nd International Conference on Information Systems Security and Privacy(ICISSP)*, pp. 407-414, 2016.

- [59] W. Wang et al. « *HAST-IDS: Learning Hierarchical SpatialTemporal Features Using Deep Neural Networks to Improve Intrusion Detection* », IEEE Access, vol. 6, 2018, pp.1792–1806.
- [60] M. Lopez-Martin et al. « *Network Traffic Classifier with Convolutional and Recurrent Neural Networks for Internet of Things* », IEEE Access, vol. 5, 2017, pp. 18,042–50.
- [61] Kaiming,He.Xiangyu,Zhang.Shaoqing,Ren.and Jian,Sun.(25 Jul, 2016). « *Identity Mappings in Deep Residual Networks* ».
- [62] SainingX.Ross, Girshick2.Piotr.Dollar, 2.Zhuowen, Tu1.Kaiming, He2.(11Apr, 2017). « *Aggregated Residual Transformations for Deep Neural Networks* ».
- [63] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.10 Dec, 2015. « *Deep residual learning for image recognition* ». In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778.
- [64] VPN-nonVPN Dataset (ISCXVPN 2016). EST.1785 UNB.  
<https://www.unb.ca/cic/datasets/vpn.html> . Consulté en Mars 2020.
- [65] pandas: powerful Python data analysis toolkit (Nov 09, 2019).  
<https://pandas.pydata.org/docs/> .Consulté en Mars 2020.
- [66] GDCoder : Mlxtend : Feature Selection. <https://gdccoder.com/mlxtend-feature-selection-tutorial/> . Consulté en Mars 2020.
- [67] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). « *Scikit-learn: Machine learning in Python. Journal of machine learning research* », 12(Oct), 2825- 2830.
- [68] NumPy Introduction. [https://www.w3schools.com/python/numpy\\_intro.asp](https://www.w3schools.com/python/numpy_intro.asp) . Consulté en Mars 2020.
- [69] New in Matplotlib.[https://matplotlib.org/2.1.2/users/whats\\_new.html#new-in-matplotlib-2-1](https://matplotlib.org/2.1.2/users/whats_new.html#new-in-matplotlib-2-1) . Consulté en Avril 2020.
- [70] Krishna.Rungta.(2018). « *TensorFlow in 1 Day: Make your own Neural Network : TensorFlow* ».449.
- [71] Basien L. (18 dec, 2018). « *Confusion matrix : l’outil de mesure de performances du machine learning* ».



