REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : RTIC13/M2/2021

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Réseaux et Technologies de l'Information et de la Communication (RTIC)**

Une gestion des tranches de réseau 5G basée sur l'apprentissage automatique

---

Par :

Achouri Amira Nouha

Soutenu le 06/07/2021 devant le jury composé de :

Bitam Salim	Prof	Président
Dr.Ayad Soheyb	**MCA**	**Rapporteur**
Bendahmane Toufik	MAA	Examinateur

Année universitaire 2020-2021

Remerciement

Je remercie dieu le tout puissant de ma donnée la santé et la volonté d'entamer ce mémoire.

Tout d'abord, ce travail ne serait pas aussi riche et n'aurait pas pu avoir le jour sans l'aide et l'encadrement de ***Dr.Ayad Soheyb,*** *je le remercie pour la qualité de son encadrement exceptionnel, pour sa patience, sa rigueur et surtout sa disponibilité.*

Je tiens aussi à remercier les membres du jury pour avoir accepté d'examiner et d'évaluer ce travail.

Ma reconnaissance va aussi à tous ceux qui ont collaboré à notre formation en particulier les enseignants du département d'Informatique, Université Mohamed Khider- Biskra.

Dédicaces

*A ma chère **mère***

Qui est partenaire à part entière de ma réussite, par son amour, son optimisme sans faille, ces longues années de sacrifices pour m'aider à avancer dans la vie et aller au delà de mes capacités. Votre présence à mes côtés a toujours été ma source de force pour faire face aux obstacles et difficultés.

*A mon cher **père***

Grâce à toi papa j'ai appris le sens du travail et de la responsabilité. Je voudrais te remercier pour ton amour, ta générosité, ta compréhension... Ton soutien fut une lumière dans tout mon parcours. Aucune dédicace ne saurait exprimer l'amour l'estime et le respect que j'ai toujours eu pour toi.

*A ma chère sœur **Chouhaz***

*A mes chers frères **Houssam & Moncef***

pour leurs encouragements permanents, et leurs soutient moral,

*A mon neveu **Arkane***
Tu as apporté beaucoup de bonheur à notre famille. Je t'aime

A toute ma famille pour leur soutien tout au long de mon parcours universitaire

Aux personnes qui m'ont toujours aidé et encouragé, qui étaient toujours à mes côtés.

*A ma chère **Soraya** et tous mes amies*
En témoignage de l'amitié qui nous unit et des souvenirs de tous les moments que nous avons passés ensemble, je vous dédie ce travail et je vous souhaite une vie pleine de santé et de bonheur.

Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fuit de votre soutien infaillible.

Table de matière

Abstract

5G, is the fifth generation of mobile telephony that succeeds 4G, it base mainly on performance indicators (KPIs) according to three use cases (enhanced mobile broadband (eMBB), massive machine type communication (mMTC) and ultra-reliable and low latency communication (URLLC)).

Since each of these use cases requires different types of network capabilities, so they cannot be efficiently provided over a single, homogeneous network.

Network slicing considered as a key technology that enables efficient deployment of multiple network slices on a single physical infrastructure to serve these use cases with the required quality of service.

The growth of data and the huge number of new devices connected to the 5G network requires the adoption of new techniques for better management of services.

In this context, we have proposed models based on machine learning techniques and deep learning to predict the appropriate slice to an incoming traffic using Key Performance Indicators (KPIs).

According to the obtained results, the deep learning model performed better than the best one of machine learning models, which is Random Forest, and this was due to the characteristics deep learning models.

Key words - 5G, Network Slicing, Machine Learning, Deep Learning.

Résumé

La 5G, est la cinquième génération de la téléphonie mobile qui succède à la 4G**,** elle est fondée principalement en fonction des indicateurs de performance (KPIs) selon trois catégories d'usages (le haut débit mobile évolué (eMBB), la communication de type machine massive (mMTC) et la communication ultra-fiable et à faible latence (uRLLC)).

Étant donné que chacun de ces catégories nécessite différents types de KPIs, ils ne peuvent pas être fournis efficacement sur un seul réseau homogène.

Le découpage de réseau est considéré comme une technologie clé qui permet de déployer efficacement plusieurs tranches du réseau virtuelles sur une infrastructure physique commune pour servir ces catégories d'usages avec la qualité de service requise.

La croissance des données et l'énorme nombre de nouveaux appareils connectés au réseau 5G, nécessite l'adoption de nouvelles techniques pour une meilleure gestion de trafic réseaux.

Dans ce contexte, nous avons proposé des modèles basés sur les techniques d'apprentissage automatique et d'apprentissage profond afin d'automatiser la gestion de trafic et prédire la tranche de réseau virtuelle approprié pour n'importe quel type d'équipement connecté dépendamment de performances applicatives requises (KPIs).

Selon les résultats obtenus, Parmi les modèles d'apprentissage automatique utilisé le Random Forest a donné de meilleurs résultats et le modèle d'apprentissage profond DNN a donné encore une meilleure performance par rapport à ce dernier et cela dû aux caractéristiques et puissance des modèles d'apprentissage profond.

Mots clés - 5G, Découpage de réseau, Apprentissage automatique, Apprentissage profond.

الملخص

تكنولوجيا 5G هو الجيل الخامس من الهواتف المحمولة الذي يخلف 4G، وهو يعتمد بشكل أساسي على مؤشرات الأداء (KPIs) وفقًا لثلاث فئات من الاستخدام (النطاق العريض المتنقل(emBB)، اتصالات كثيفة من نوع الاتصالات الآلية (mMTC)والاتصالات فائقة الموثوقية أو منخفضة الكمون (uRLLC)).

نظرا أن كل فئة من هذه الفئات لها متطلباتها الخاصة فلا يمكن توفيرها بطريقة فعالة عبر شبكة واحدة متجانسة لهذا يجب تقسيم الشبكة التقنية الأساسية لتحقيق هذا الأخير، تتم إدارة مختلف قطع شبكة 5Gوالتنبؤ بها باستخدام مؤشرات الأداء (KPIs) .

ان تزايد ا لبيانات والعدد الهائل من الأجهزة الجديدة المتصلة بشبكة 5G تتطلب اعتماد تقنيات جديدة لتحسين إدارة مختلف الخدمات المقدمة.

في هذا المشروع نقترح نماذج تستند على التعلم الآلي والتعلم العميق للتنبؤ بالقطعة (slice) المناسبة لأي نوع من المعدات المتصلة بالشبكة اعتمادا على الأداء الذي يتطلبه (KPIS).

وفقًا للنتائج التي تم الحصول عليها، كان أداء نموذج التعلم العميق(DNN) أفضل من نموذج التعلم الآلي(RF) وهذا يرجع إلى خصائص هذا الأخير.

الكلمات المفتاحية - 5G ، تقسيم الشبكة ، التعلم الآلي ، التعلم العميق.

Liste des figures

Liste des tableaux

Chapitre 1

Introduction Générale

1.1.Contexte de travail

La 5G est la cinquième génération des réseaux mobiles qui est censé de prendre en charge divers nouveaux cas d'utilisation (les véhicules autonome, téléchirurgie, les villes intelligentes…etc.), qui peuvent être classés selon les huit indicateurs de performance (KPIs-Key Performance Indicators) en trois catégories qui sont le haut débit mobile évolué (eMBB), les communications massives de type machine (mMTC), et les communications ultra-fiables et à faible latence (URLLC).

Le défi de la 5G est d'assurer la performance du réseau et les différentes exigences de qualité de service (QoS) de différentes catégories d'usage. Où chacun de ces derniers nécessite différents indicateurs de performances (par exemple, la catégorie d'usage eMBB nécessite une très grande largeur de bande et la catégorie d'usage mMTC exige une connectivité ultra-dense). [1]

Le projet de partenariat de troisième génération (3GPP[1]) considère le découpage du réseau comme une technologie clé pour la 5G.

Le découpage permettrait aux opérateurs d'exécuter efficacement plusieurs instances du réseau sur une seule infrastructure physique pour servir les diverses cas d'utilisation des catégories d'usage définit, avec une bonne qualité de service (QoS) [2], en mettant en œuvre des tranches de réseau une pour chaque catégorie d'usage. À leur tour, différents services seront associés aux différentes tranches en fonction de leurs besoins et de leurs KPIs. Chaque tranche (slice) sera configurée en termes de topologie, de capacité et de services afin de satisfaire les exigences d'une ou de plusieurs catégories spécifiques (eMBB, mMTC et uRLLC). [3]

Le secteur des télécommunications connaît une transformation numérique massive avec l'adoption des algorithmes du Machine Learning et Deep Learning qui sont actuellement mis en œuvre dans diverses industries. [2]

Avec l'augmentation des données de nature complexe et l'immense volume d'informations sur la 5G, la gestion d'une tranche de réseau afin qu'elle puisse satisfaire

[1] La 3GPP est une coopération entre organismes de normalisation en télécommunications.

efficacement la qualité de service nécessite l'automatisation, les techniques d'apprentissage automatique et d'apprentissage profond facilitent cette automatisation.

1.2. Problématique et objectifs

Etant donné que le découpage de réseau permettrait de fournir efficacement plusieurs tranches du réseau sur une seule infrastructure pour servir diverses catégories d'usage en assurant la qualité de service (QoS), beaucoup de défis peuvent être soulevés :

- La sélection appropriée de la tranche pour un utilisateur en fonction du trafic entrant,
- La surveillance de tranche basée sur la prédiction du trafic….etc.

Dans le cadre de ce Projet , nous avons développé des modèles basée sur les techniques d'apprentissage automatique et d'apprentissage profond pour la prédiction de la tranche de réseau appropriée pour n'importe quel type d'équipement connecté dépendamment de performances applicatives requises (KPIs).

1.3. Contributions

Afin d'atteindre nos objectifs nous avons proposé deux contributions :

(i) La première contribution consiste à l'analyse et le traitement de l'ensemble de données comme suit :

- La première étape consiste à résoudre le **problème de déséquilibres de distribution de classes** ; la notion de déséquilibre de classe peut être définit comme étant les proportions des différentes classes ne sont pas strictement identiques. Pour faire face à ce déséquilibre de classes nous avons utilisés la génération de données synthétiques en utilisant l'algorithme de SMOTE (Synthetic Minority Over-sampling Technique) qui consiste à sur-échantillonner en se basant sur les proches voisins de la classe minoritaire.
- La deuxième étape est l'étape de prétraitement de l'ensemble de données pour qu'on puisse à l'utilisée pour la classification des données avec les techniques choisis.

(ii) La deuxième contribution concerne le bon choix des techniques a utilisées pour répondre à telle problématique et qui correspond avec la nature d'ensemble de donnée utilisée. Nous avons choisis des algorithmes d'apprentissage automatique et un réseau de

neurones profond afin d'étudier le meilleur résultat en utilisant les déférents métriques d'évaluation de classification (précision, rappel, la mesure-f, AUC, la statique Cohens kappa, taux d'exactitude).

1.4.Structure de mémoire

Le présent mémoire est organisé en cinq chapitres dont les thèmes sont donnés ci-dessous :

- **Le premier chapitre** présente le contexte et la problématique de ce travail. Ce chapitre identifie également l'objectif de ce mémoire et présente brièvement les principales contributions.
- **Le deuxième chapitre est** consacré à l'étude générale de la 5éme génération des réseaux mobiles, Ce chapitre réuni deux parties principale, la première partie est consacrée à l'étude générale de la 5G.La seconde partie englobe le concept de découpage de réseau 5G.
- **Le troisième chapitre** fournit une compréhension sur les principes d'apprentissage automatique et les réseaux de neurones profonds en exprimant leurs rôles pour l'automatisation des taches de découpage de réseau 5G, par la suite nous avons présenté les travaux récemment réalisées et qui traite la même problématique.
- **Le troisième chapitre** nous avons détaillé dans ce chapitre les différentes étapes et méthodes utilisées pour le traitement de la problématique.
- **Le quatrième chapitre** montre l'implémentation des différentes étapes décrites au précédant chapitre. En commençant par la présentation des outils et l'environnement de développement, ensuite, nous avons présenté les résultats obtenus après l'implantation de nos modèles. Nous terminons par une étude comparative.

Dans la conclusion, nous résumons et passons en revue nos idées et résultats en donnant des améliorations possibles et quelques perspectives.

Chapitre 2

Des généralités sur la 5G

2.1. Introduction

La 5éme génération des réseaux mobiles a promet des améliorations de performance sans précédent en termes de débit , latence et d'efficacité énergétique par rapport à la 4G , un aspect important pour atteindre cet objectif est le découpage du réseau qui vise à servir de multiples réseaux logiques virtualisés et indépendants sur la même infrastructure de réseau physique de la 5G.

Dans ce chapitre, nous allons présenter en premier lieu l'architecture de réseau 5G. Ensuite, nous passons à la présentation des différentes catégories d'usage de ce futur système de communication. Nous détaillons par la suite les différents indicateurs de performances et les technologies envisageables pour répondre aux attentes de la 5G. Enfin, nous s'intéressons particulièrement à détaillée le concept de découpage de réseau 5G.

2.2. La 5G

Depuis le vrai premier appel téléphonique mobile, il y a 44 ans, les technologies mobiles ont évolué de manière continue ainsi que leurs performances de manière exponentielle : le service voix, puis la messagerie et enfin l'internet mobile, l'évolution des réseaux mobiles et le passage d'une génération à l'autre.

La naissance de la technologie LTE et de la quatrième génération (4G), couplée à la démocratisation des smartphones et tablettes, a mené à une augmentation très forte des volumes de données échangés en mobilité. L'utilisation d'un mobile et de ses applications est désormais fermement ancrée dans le quotidien de nos concitoyens : les appareils portables connectés sont de plus en plus performants : ils remplacent bien souvent le téléphone fixe, l'appareil photo, voire l'ordinateur et même le téléviseur.

Aujourd'hui, des millions de vidéos sont vues sur YouTube et des images sont chargées sur Instagram chaque minute.

Le dernier rapport d'Ericsson indique que, en l'espace d'un an, le volume de données échangées sur les réseaux mobiles a presque doublé et que dans 5 ans il aura été multiplié par 10 par rapport à l'utilisation actuelle. De nouvelles solutions doivent donc être trouvées afin de pouvoir répondre à cette demande et d'optimiser l'utilisation des ressources. L'augmentation du nombre d'applications, leur diversification ainsi que l'amélioration de la

qualité des réseaux mobiles ont conduit à l'augmentation de la demande, à l'apparition de nouveaux usages (objets connectés, drones, etc…) et de nouveaux utilisateurs.

La 5G se situe au carrefour de ces nouveaux usages ; elle ambitionne de répondre mieux et simultanément à cette grande variété de besoins et ces nouvelles demandes, via une technologie unifiée qui prend en compte cette diversité dès sa conception. [4]

2.3. Architecture de réseau 5G

Le réseau 5G (5G System) se compose d'un accès Radio (NG-RAN : Next Generation Radio Access Network) et d'un cœur réseau (5G Core). (voir figure 2.1)[5]

2.3.1. L'accès radio 5G (RAN)

L'accès radio 5G est constitué de stations de base de nouvelle génération qui forment le nœud de connexion des mobiles avec le cœur réseau 5G (5GC)

2.3.2. Le cœur réseau 5G (5GC*)*

Le cœur réseau 5G est adapté pour la virtualisation du réseau et s'appuie sur le découpage du plan de contrôle (Control Plane) et du plan utilisateur (User Plane).

Le cœur de réseau 5G est composé principalement de :

- ✓ L'entité AMF (Access and Mobility Managmenent Function) établit une connexion NAS[1] avec l'équipement d'utilisateur (User Equipement – UE) et a pour rôle l'attachement des UE.

- ✓ L'entité SMF (Session Management Funtion**)** reprend le rôle de l'entité SGW-C et PGW-C. L'entité SMF permet de contrôler les sessions PDN. L'entité SMF est choisie par l'entité AMF puisque l'entité AMF gère la signalisation NAS avec le mobile. L'entité SMF est responsable de la gestion du plan de contrôle. L'entité SMF a une interface avec l'entité qui gère la politique des flux (PCF : Policy Charging Function).

- ✓ L'entité PCF permet de contrôler les flux à la fois au niveau de l'entité SMF mais également au niveau de l'entité AMF afin de pouvoir apporter une meilleure granularité sur les flux autorisés en prenant en compte la localisation du mobile UE.

[1] NAS (Non Access Stratum) représente un ensemble de protocoles qui s'établit entre l'UE et le réseau cœur.

- ✓ Le profil utilisateur (son abonnement, ses droits, …) sont sauvegardées dans une base de données UDR accessible via l'entité UDM (Unified Data Management). L'entité UDM conserve les profils de sessions de données (sessions PDU) et de l'entité AMF sur laquelle est attachée le mobile UE.

- ✓ L'enregistrement du mobile nécessite une double authentification réalisée au niveau de l'entité AMF et du mobile UE à partir de vecteurs d'authentifications fournies par l'entité AUSF (AUthentication Server Function).

- ✓ Enfin, l'entité NSSF (Network Slice Selection Function) est une entité permettant d'assister l'entité AMF de la sélection des instances logiques du réseau pour une tranche de réseau (slice) défini.

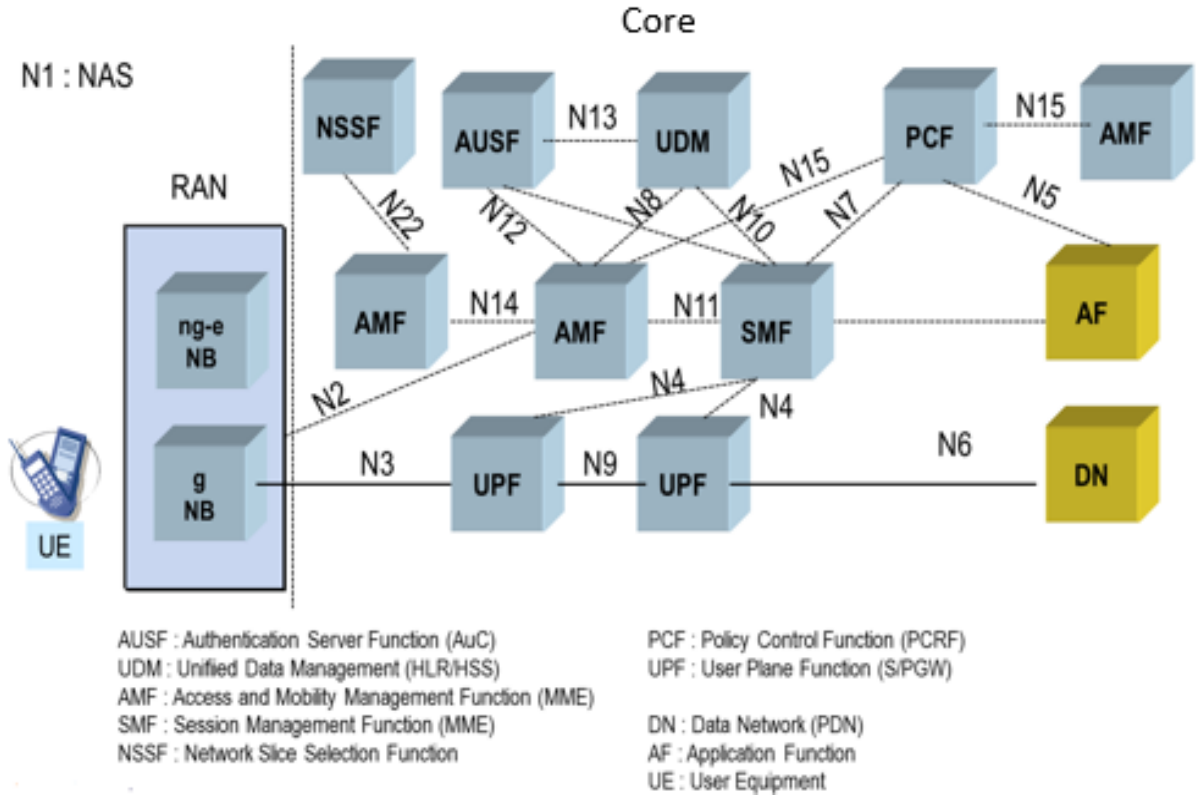


Figure 2.1: L'architecture du réseau 5G point à point (R.15). [5]

2.4. Les catégories d'usages de la 5G

Trois catégories d'usages sont définies par l'UIT[2], sous le terme IMT- 2020[3] : [6] [7]

2.4.1. Les communications massives de type machine (mMTC – Massive Machine Type Communications)

Les communications entre une grande quantité d'objets avec des besoins de qualité de services variés. L'objectif de cette catégorie est de répondre à l'augmentation exponentielle de la densité d'objets connectés ;

2.4.2. La large bande mobile évolué (eMBB – Enhanced Mobile Broadband)

La connexion en ultra haut débit en outdoor et en indoor avec uniformité de la qualité de service, même en bordure de cellule ;

2.4.3. Les communications ultra-fiables à très faible latence (uRLLC – Ultra-reliable and Low Latency Communications)

Les communications ultra-fiables pour les besoins critiques avec une très faible latence, pour une réactivité accrue.

La Figure 2.2 présente le résumé des trois cas d'utilisation de la 5G dans les différents domaines d'applications.

[2] L'Union internationale des télécommunications ou UIT est l'agence des Nations unies pour le développement spécialisé dans les technologies de l'information et de la communication.
[3] Les télécommunications mobiles internationales-2020 sont les exigences émises par le Secteur des radiocommunications de l'UIT de l'Union internationale des télécommunications en 2015 pour les réseaux, appareils et services 5G.

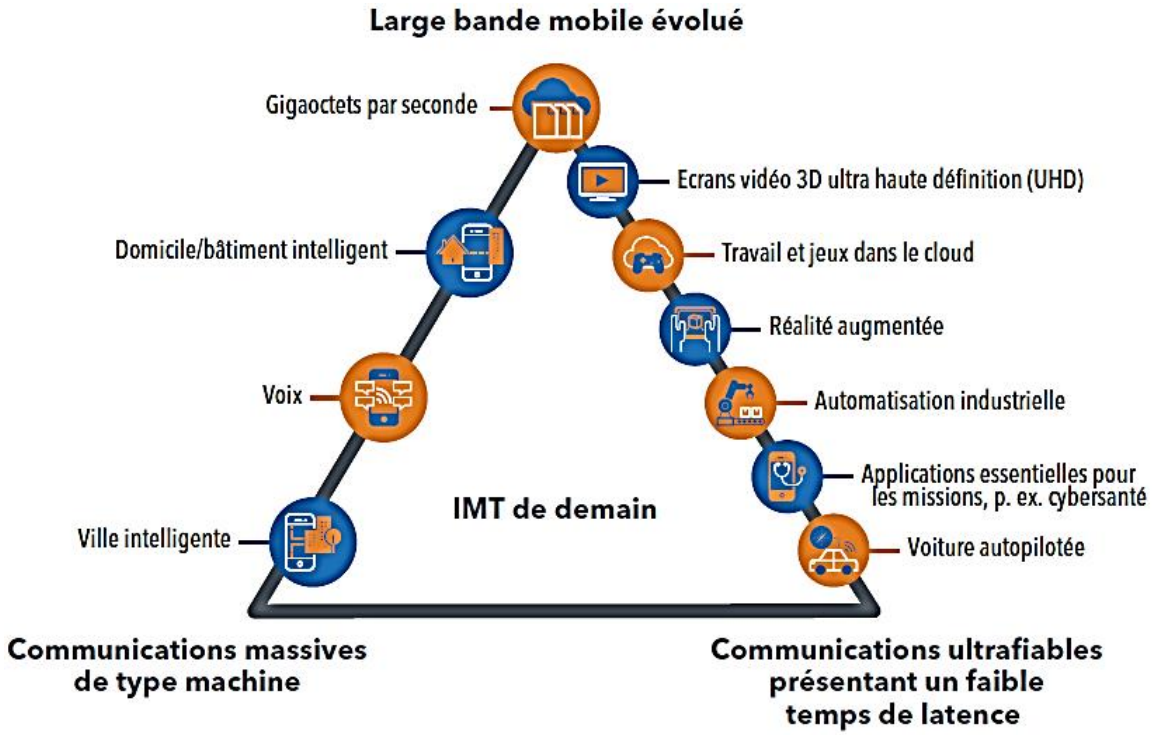


Figure 2.2 : Catégorie d'usages de 5G. [4]

- La première catégorie d'usages (mMTC) englobe principalement tous les usages liés à l'Internet des objets. Ces services nécessitent une couverture étendue, une faible consommation énergétique et des débits relativement restreints. L'apport annoncé de la 5G par rapport aux technologies actuelles réside dans sa capacité à connecter des objets répartis de manière très dense sur le territoire. [8]

- La large bande mobile évolué (eMBB) concerne tous les applications et services qui nécessitent une connexion toujours plus rapide, pour permettre par exemple de visionner des vidéos en ultra haute définition (8K) ou de « streamer » sans-fil des applications de réalité virtuelle ou augmentée. [8]

- Les communications ultra-fiables à très faible latence (uRLLC) regroupent toutes les applications nécessitant une réactivité extrêmement importante ainsi qu'une garantie très forte de transmission du message. Ces besoins se retrouvent principalement dans les transports (temps de réaction en cas de risque d'accident, par exemple), dans la médecine (téléchirurgie) et, de manière générale, pour la numérisation de l'industrie. [8]

2.5. Les indicateurs de performances de la 5G

Afin de mettre en œuvre ces trois types d'usages, huit indicateurs de performance (en anglais KPI – Key Performance Indicators) ont été établis par l'UIT1 pour préciser, quantifier et mesurer les caractéristiques de systèmes IMT-2020 (5G) : [9][4][7]

1) Débit de données maximale

Le dédit de données maximale est définie comme étant le débit de données maximal obtenu dans des conditions idéales par utilisateur/appareil (mesuré en Gbit/s).

2) Débit moyen perçu par l'utilisateur

Le débit de données par utilisateur est défini comme étant le débit moyen de l'utilisateur c'est à dire le nombre de bits correctement reçus par les utilisateurs. La valeur moyenne des débits par utilisateur varie selon les zones géographiques :

➢ Des débits +100 Mbps devraient généralement être réalisables dans les environnements urbains et suburbains.

➢ Des débits de données d'au moins 10 Mbps devraient être accessibles presque partout, y compris dans les zones rurales peu peuplées des pays développés et des pays en développement.

3) Latence

Cette exigence est définie comme étant le temps nécessaire à un paquet de données pour passer de la source à la destination à travers un réseau (mesuré en ms).

4) Mobilité

Vitesse maximale à laquelle une QoS défini un transfert continu entre les nœuds radio pouvant appartenir à différentes couches et/ou technologies d'accès radio (multicouche/-RAT) peuvent être obtenus (mesuré en km/h).

5) Densité de raccordement

La densité de raccordement est la capacité à soutenir la livraison réussie d'un message d'une certaine taille dans un laps de temps, même dans des endroits très denses comme une

gare, un stade de football, etc. Contrairement à la 4G qui n'offrait que des milliers de connexions par kilomètres carré, la 5G va multiplier ce chiffre par un facteur dix afin d'atteindre un million de connexions par kilomètre carré (mesuré en quantité d'objets/km²).

6) Efficacité énergétique des réseaux

L'efficacité énergétique est définie comme étant le nombre de bits transmis par Joule d'énergie et elle comporte deux aspects :

- **Côté réseau,** l'efficacité énergétique correspond à la quantité de bits d'information reçue ou transmise par les utilisateurs, par unité de consommation d'énergie du réseau d'accès radioélectrique (RAN) (mesuré en bit/joule).
- **Côté dispositif,** l'efficacité énergétique correspond à la quantité de bits d'information par unité de consommation d'énergie du module de communication (mesuré en bit/joule).

7) Efficacité du spectre

L'Efficacité spectrale est définie comme étant la quantité de données binaires (bits) pouvant être transférée durant une seconde sur une largeur de bande de 1 Hz (mesuré en bit / s / Hz / cellule).

8) Capacité de trafic d'une zone

La capacité de trafic d'une zone est définie comme étant le débit total de trafic fourni par zone géographique (mesuré en Mbit/s/m2).

Les principales capacités de l'IMT-2020 (5G) sont illustrées à la Figure 0.3 par rapport à celles de l'IMT-Evoluées (4G).

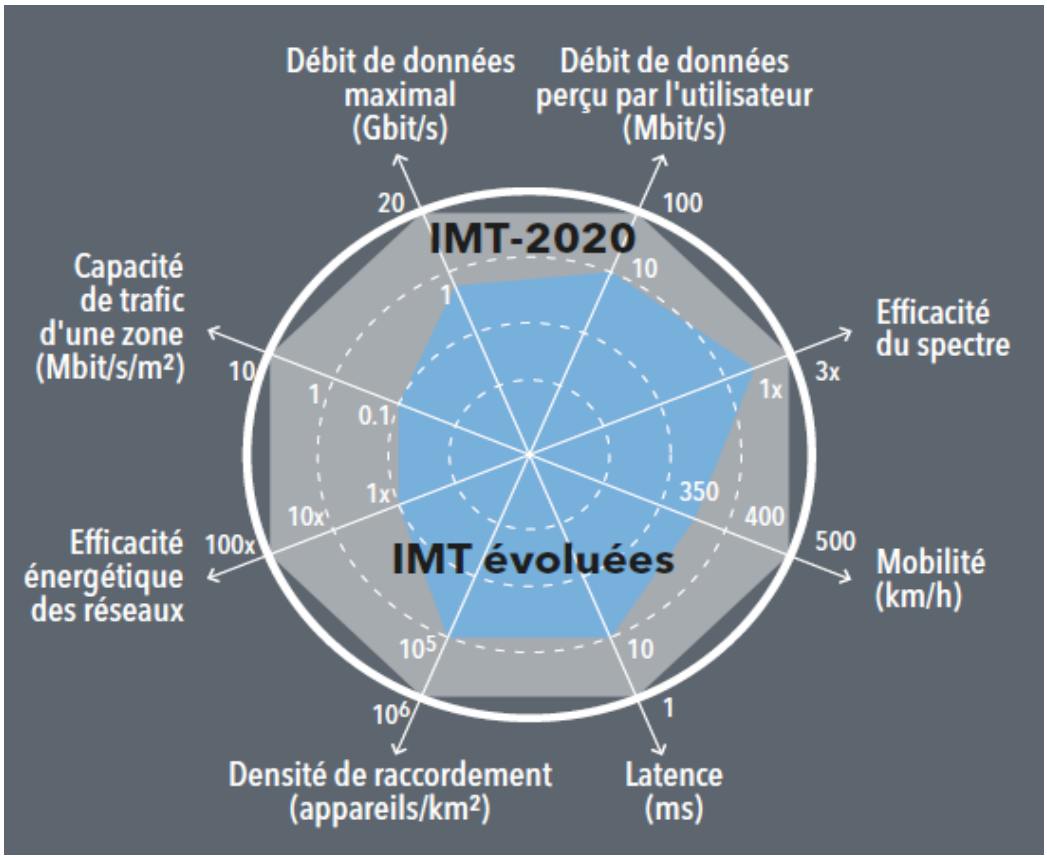


Figure 2.3: Comparaisons entre 4G et de la 5G au niveau des huit indicateurs de performance. [10]

Il est fondamental de comprendre que l'ensemble des indicateurs présentés en 2.3 détermine l'enveloppe des performances maximales de la 5G. Cependant, ces valeurs extrêmes ne pourront être atteintes simultanément pour tous les indicateurs : [4]

Tous les besoins ou catégories d'usage ne sont pas compatibles entre eux et un choix devra être réalisé pour définir des classes d'utilisation disposant chacune de son enveloppe de performances, notamment pour les trois catégories d'usages décrites dans la section 2.3 (mMTC, eMBB et uRLLC).

C'est le principe de découpage de réseau ou chaque « tranche » dispose de son enveloppe qui est un compromis lié à l'usage ciblé ; à l'intérieur d'un réseau 5G, les caractéristiques devront s'adapter à l'environnement choisi. La figure 2.4 positionne, sur l'étoile des 8 KPI susmentionnés, les trois catégories d'usage principales.

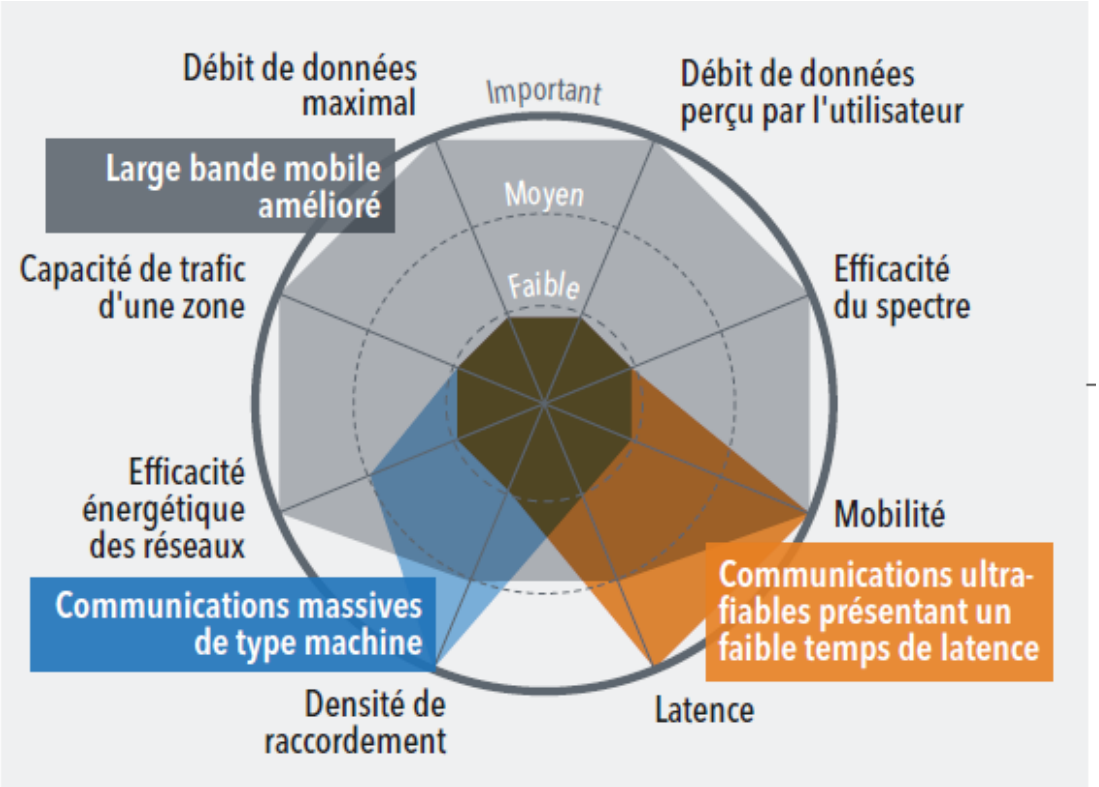


Figure 2.4: Les indicateurs clés de performance pour les trois catégories d'usage de la 5G. [10]

2.6. Les technologies émergentes de la 5G

Afin d'atteindre les objectifs et les visions ambitieuses mentionnés, plusieurs technologies clés ont été identifiées.

2.6.1. Les ondes millimétriques

L'utilisation de bandes millimétriques (en anglais mmWaves) constitue l'une des technologies de rupture de la 5G. Cette appellation correspond aux fréquences supérieures à 6 GHz qui n'ont encore jamais été prises en compte pour le déploiement des réseaux mobiles (fronthaul) pour des raisons de maturité technologique et de qualité de propagation. Pour répondre à l'incessante augmentation des débits et des volumes de données échangés, il est nécessaire d'utiliser de nouvelles bandes disposant de très larges canalisations (plus de 100 MHz par utilisateur).

Les bandes millimétriques pourraient offrir de telles réserves de spectre et leur utilisation permettrait d'atteindre les débits de la 5G.

En contrepartie, leur utilisation impose le développement de toutes les technologies

nécessaires, miniaturisées, à bas coût et avec une consommation énergétique compatible avec des terminaux portables (amplificateurs, codeurs, traitement de signal, antennes…). En particulier, à cause de la faible qualité de propagation des ondes millimétriques, chaque cellule aura une couverture réduite, ce qui nécessitera la mise en place de techniques de beamforming, décrites ci-dessous pour mieux focaliser l'énergie transmise par les antennes. [4][11]

2.6.2. Les petites cellules

Les petites cellules (en anglais Small cells) sont d'infimes stations de base de faible puissance qui peuvent être placées à moins de 100 m de distance pour couvrir de petites zones géographiques. Ces stations de base de faible puissance empêchent le signal de chuter dans les zones surpeuplées.

Les Small cells sont très légères et petites ; ainsi, ils peuvent être placés n'importe où. Si nous utilisons des ondes millimétriques au lieu du spectre traditionnel inférieur à 6 GHz, la petite cellule peut devenir encore plus petite et peut être installée dans des endroits minuscules. Les petites cellules joueront un rôle important dans la fourniture d'un haut débit et d'une latence ultra-faible pour la 5G. [12]

2.6.3. MIMO Massive (Multiple Inputs - Multiple Outputs)

Cette technologie se caractérise par l'utilisation d'un nombre élevé de micro antennes « intelligents », situées sur le même panneau (de 8 à 128 actuellement, mais le nombre augmentera avec l'utilisation de fréquences supérieures à 6 GHz).

L'intérêt d'utilisation du massive MIMO est :

- d'une part, cette technologie permet d'augmenter les débits, grâce au multiplexage spatiotemporel ;
- d'autre part, elle permet de focaliser l'énergie sur un terminal, pour améliorer son bilan de liaison, grâce à la formation de faisceau, ou beamforming. [4]

2.6.4. Formation de faisceau

La formation de faisceau (en anglais Beamforming) présente plusieurs avantages pour les réseaux 5G.

Pour les systèmes MIMO massifs, le Beamforming aide à augmenter l'efficacité du spectre, et pour les ondes millimétriques, elle aide à augmenter le débit de données.

Dans les systèmes MIMO massifs, la station de base peut envoyer des données à l'utilisateur depuis différents chemins, et le Beamforming chorégraphie le mouvement du paquet et l'heure d'arrivée pour permettre à plusieurs utilisateurs d'envoyer des données simultanément comme le montre la figure 2.5.

Comme les ondes millimétriques ne peuvent pas pénétrer à travers les obstacles et ne se propagent pas à de plus longues distances car la longueur d'onde est courte, la formation de faisceau aide ici à envoyer des faisceaux concentrés vers les utilisateurs. Ainsi, la formation de faisceau aide un utilisateur à recevoir un signal fort sans interférence avec d'autres utilisateurs. [13]

Figure 2.5: Massive Multiple Output–Multiple Output (MIMO) Beamforming. [13]

2.6.5. Technologie Full Duplex

Dans les systèmes classiques, l'émission et la réception se font soit sur des bandes de fréquences différentes (duplexage en fréquences dit FDD-Frequency Division Duplexing) soit à des instants différents (duplexage temporel dit TDD-Time Division Duplexing). Le full duplex ambitionne de permettre l'émission et la réception simultanée d'information, sur les mêmes fréquences, au même moment et au même endroit.(voir figure 2.6) [13][4]

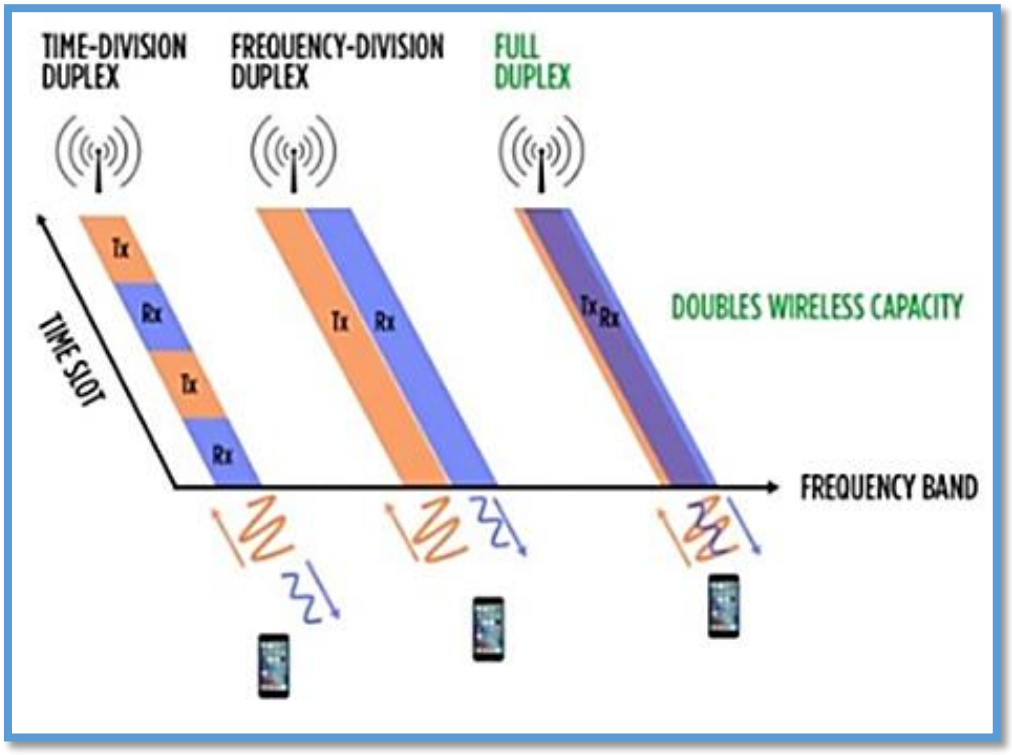


Figure 2.6: Illustration de répartition Full Duplex, comparé au FDD et TDD. [7]

2.6.6. Multiplexage NOMA (NOMA-Non Orthogonal Multiple Access)

Dans un système d'accès multiple orthogonal (OMA) l'allocation des ressources est utilisée par les utilisateurs pour éviter les interférences intra-cellules (inter-utilisateurs), par contre pour améliorer l'efficacité spectrale de la 5G, par rapport à la 4G, des méthodes de multiplexage non orthogonales, c'est-à-dire que plusieurs utilisateurs peuvent recourir aux mêmes fréquences au même moment. [12]

2.6.7. Contrôleur de réseau logiciel (SDN) et virtualisation du réseau (NFV)

A. Contrôleur de réseau logiciel (SDN -Software Defined Networking)

SDN est un paradigme de mise en réseau qui promet d'améliorer la programmabilité et la flexibilité des réseaux qui a pour objectif de dissocier la partie contrôle d'un réseau de sa partie opérationnelle, ces deux parties étant traditionnellement liées et distribuées (de manière figée) dans le réseau. Le contrôle du réseau, autrefois dévolu à des composants matériels spécialisés et non évolutifs, est centralisé sous forme de logiciels sur des serveurs plus puissants et affranchis (en théorie) des spécifications des équipementiers. Cela permet

le déploiement de services à forte valeur ajoutée (équilibrage de charge, routage intelligent, configuration dynamique…etc.) dans des environnements hétérogènes. [4]

B. Virtualisation du réseau *(NFV -Network Function Virtualization)*

Le NFV, complémentaire du SDN, a pour objectif de virtualiser, c'est-à-dire remplacer par du logiciel sur un serveur, des équipements matériels spécialisés dans certaines fonctions clés du réseau (firewall, cœur de réseau, interfaces entre différents systèmes…), dans le but d'accélérer les déploiements et de permettre des évolutions rapides. [6]

En d'autre terme NFV sert à visualiser les fonctions réseau (y compris les fonctions de transfert et de contrôle du réseau) à partir du matériel. C'est-à-dire que les fonctions de l'équipement réseau traditionnellement dédié (par exemple, routeur, pare-feu et équilibreurs de charge) peuvent être fournies comme des fonctions logicielles. [14]

2.6.8. Découpage de réseau

Permet une « découpe » virtuelle d'un réseau de télécommunications en plusieurs tranches (slices). Cela permet de fournir des performances différentes associées à chaque tranche, et donc d'allouer des ressources dédiées par catégories d'usage ou d'objet ; par exemple en termes de fiabilité, de bande passante, de latence…etc. Chaque tranche de réseau correspond ainsi à un usage, sans empiéter sur les autres. [14]

2.7. Les concepts de découpage de réseau

2.7.1. Définition de découpage de réseau

Le découpage de réseau (ou Network Slicing en anglais) fait référence au partitionnement d'un réseau physique en plusieurs réseaux virtuels ; chaque réseau peut être personnalisé et optimisé pour un type spécifique d'usage. En tirant parti des technologies de virtualisation et de softwarisation, les ressources partagées du réseau physique peuvent être attribuées de façon dynamique et efficace en tranches de réseau logiques en fonction de différentes demandes des utilisateurs. [15]

2.7.2. Définition d'une tranche de réseau

Une tranche de réseau est un réseau logique qui fournit des capacités et des caractéristiques réseau spécifiques. Les tranches de réseau permettent la création de réseaux personnalisés pour fournir des solutions flexibles pour différents catégories d'usages qui ont des exigences diverses, en ce qui concerne les fonctionnalités, les performances et l'allocation des ressources. [16]

2.7.3. Principes de fonctionnement de découpage de réseau

Comme le montre la Figure 2.7, lorsqu'un UE s'inscrit avec un PLMN, elle doit informer le réseau avec un NSSAI demandé correspondant au type de tranche auquel l'UE s'attend à accéder, si l'UE a un NSSAI configuré ou autorisé.

- ✓ Au cours des processus d'établissement de connexion, différents UE peuvent indiquer les NSSAI avec des valeurs de type de tranche différentes en fonction des exigences des utilisateurs.
- ✓ Dès la réception du message RRC de l'UE avec une demande NSSAI, le RAN sélectionne une AMF appropriée en fonction de l'NSSAI demandée et transmet le message relatif à l'UE à l'AMF, qui peut interroger l'UDM pour récupérer les informations d'abonnement de l'utilisateur, y compris le S-NSSAI souscrit.
- ✓ L'AMF vérifie si l' NSSAI demandée est autorisée ou non en fonction de S-NSSAI souscrite.
- ✓ Compte tenu des différents abonnements, différents types de tranches de réseau peuvent être approuvés pour différents utilisateurs. Lorsque le contexte local de l'UE n'inclue pas un NSSAI autorisé, l'AMF doit interroger le NSSF.
- ✓ Le NSSF sélectionne une instance de tranche réseau approprié, y compris les fonctions de plan de commande et de réseau de plan utilisateur, et détermine l'AMF cible définie pour servir l'UE.
- ✓ Ensuite, le NSSF répond à l'AMF actuelle avec le NSSAI autorisé, qui est transmis à l'UE.
- ✓ Les règles de la politique de sélection des tranches de réseau associent une application avec une ou plusieurs tranches de réseau correspondant au S-NSSAI souscrit de l'UE.
- ✓ Lorsqu'une application associée à un S-NSSAI spécifique demande la transmission du trafic, l'UE instancie le processus d'établissement de session de l'unité de données de protocole avec les fonctions réseau du plan de contrôle de tranche.

✓ Par la suite, le trafic de données utilisateur est traité par les fonctions personnalisées du plan utilisateur de la tranche réseau. [12]

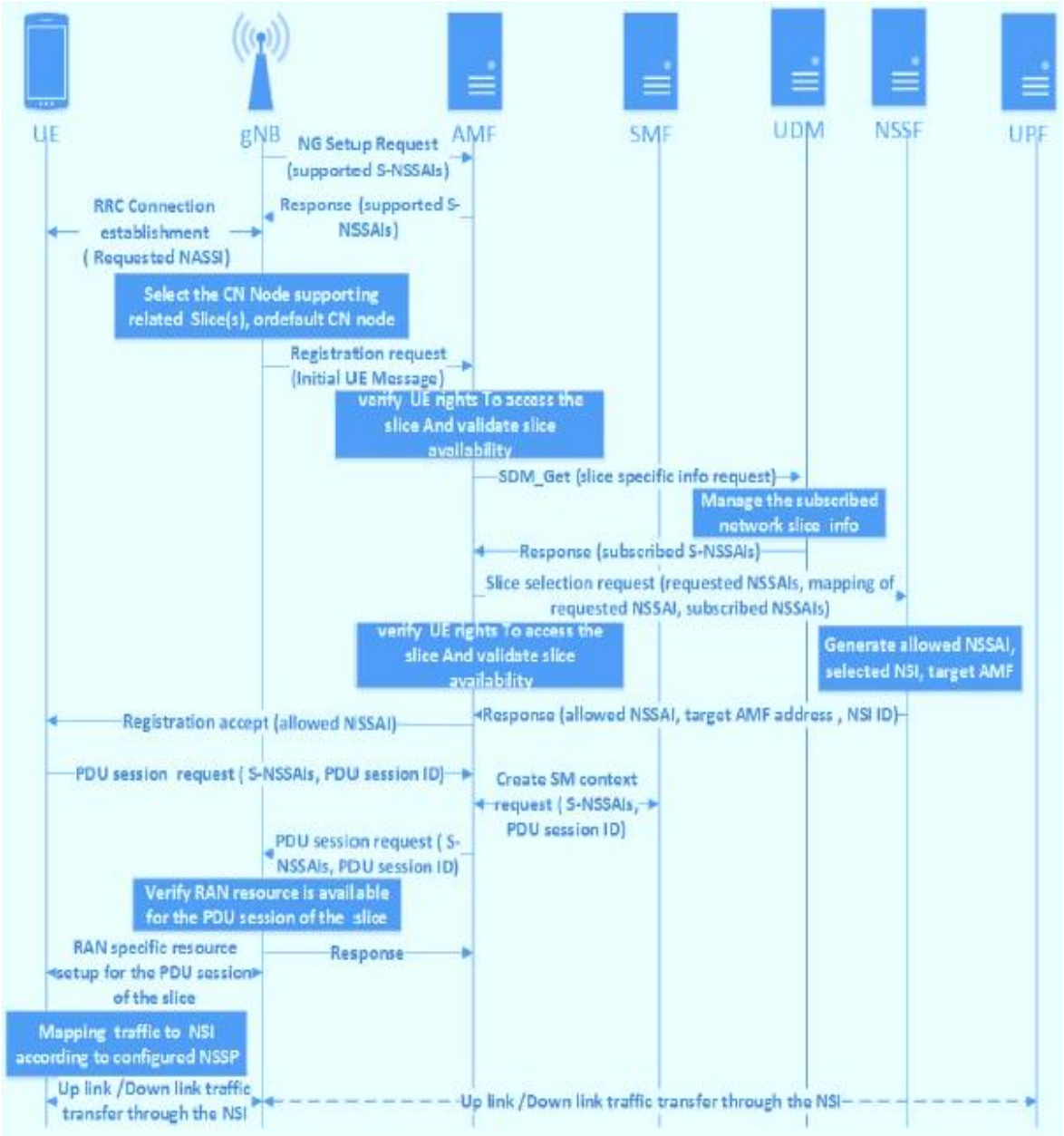


Figure 2.7: Processus de fonctionnement de découpage de réseau 5G. [15]

2.7.4. Exemple de découpage de réseau

Les besoins contrastés des différentes exigences 5G ont conduit à la conception d'une architecture de service 5G basée sur le découpage réseau 5G. Le tableau 2.1 décrit les exigences de principales catégories d'usages des réseaux mobiles 5G.

Ce tableau explique les principales catégories d'usages de la 5G et fournit quelques exemples et principales exigences associées.

En effet, la 5G envisage la conception et la mise en œuvre des trois types de tranches de réseau illustrées dans la figure ci-dessous, une pour chaque catégorie d'usage. À leur tour, différents services seront associés aux différentes tranches en fonction de leurs besoins et de leurs KPIs. Chaque tranche sera configurée en termes de topologie, de capacité et de services afin de satisfaire les exigences d'une ou de plusieurs catégories spécifiques (eMBB, mMTC et URLLC).

La conception des tranches de réseau associées aux trois catégories d'usages est illustrée à la figure 2.8. [3]

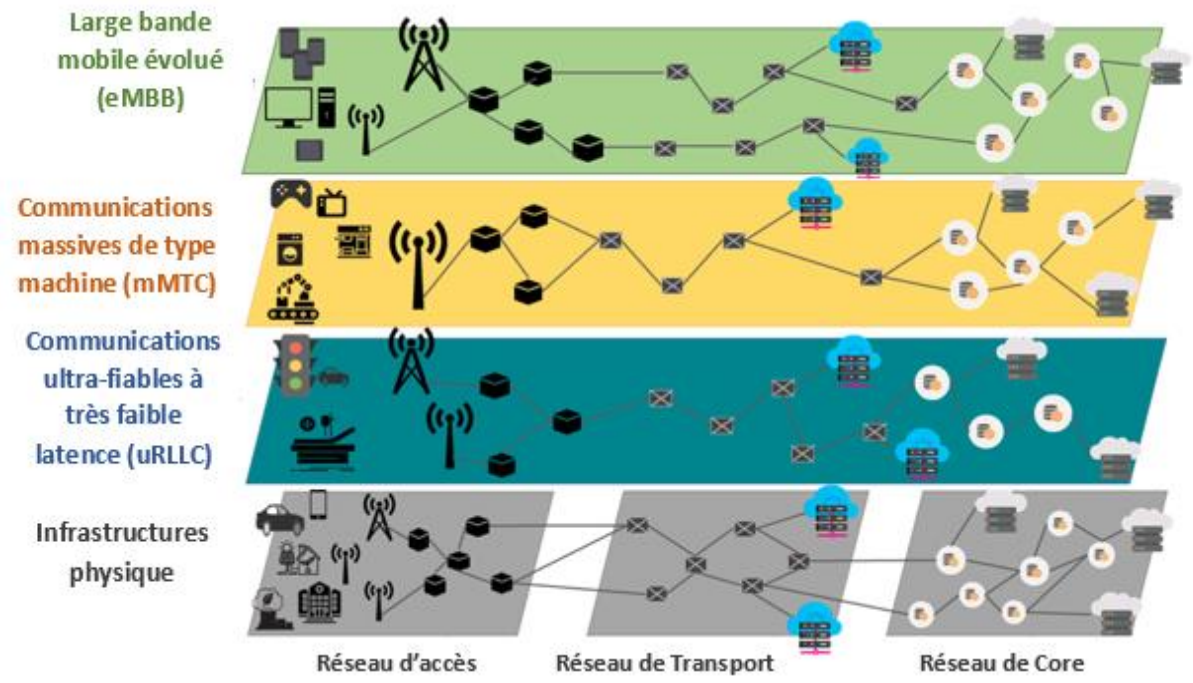


Figure 2.8: Découpage de réseau 5G. [3]

Catégories d'usages	Exemple	Exigences
eMBB	4K/8K UHD, Reality Augmenté (RA) /Réalité Virtuel (RV)	Haute capacité, cache vidéo
mMTC	Réseau de capteurs (mesure, agriculture, bâtiment, logistique, ville, maison, etc.)	Connexion massive (200 000/km2), principalement des appareils immobiles, haute énergie efficacité
uRLLC	Contrôle de mouvement, conduite autonome, usine automatisée, réseau intelligent	Faible latence (ITS 5ms, contrôle de mouvement 1 ms), haute fiabilité

Tableau 2.1: Principaux cas et exigences d'utilisation de la 5G. [3]

2.8. Conclusion

Dans le chapitre présent nous avons introduit des généralités sur la 5G ensuite nous somme concentré à détailler le principe de découpage de réseau 5G qui est considéré comme la technologie clés de la 5G.

Les réseaux de communication 5G deviennent complexes en raison de l'émergence d'un nombre sans précédent de nouveaux appareils connectés et de nouveaux types de services. En outre, les exigences de création de tranches de réseau virtuelles adaptées à fournir des services optimaux pour divers utilisateurs et applications posent des défis à la gestion efficace des ressources du réseau, l'exploitation, l'administration et la gestion d'une tranche de réseau afin qu'elle puisse satisfaire efficacement la qualité de service (QoS).Il est donc nécessaire d'automatiser ces tâches.

Les techniques d'apprentissage automatique facilitent l'automatisation des fonctions de découpage réseau.

Dans le chapitre suivant nous présenterons le rôle de l'intelligence artificielle et l'apprentissage automatique pour l'automatisation de découpage de réseau 5G .C'est à dire, l'analyse de trafic et l'attribution de la tranche adéquate selon les indicateurs de performance (KPIs).

Chapitre 3

L'apprentissage automatique dans les réseaux 5G

3.1. Introduction

L'apprentissage automatique est un type de l'intelligence artificiel qui peut fournir des solutions plus simples à des problèmes complexes en analysant un énorme volume de données en peu de temps, en apprenant à adapter ses fonctionnalités à des environnements dynamiques changeants et en prédisant les événements futurs avec une assez bonne précision.

Les réseaux de communication 5G se complexifient en raison de l'émergence d'un nombre sans précédent de nouveaux appareils connectés et de nouveaux types de services.

En outre, les exigences de création de tranches de réseau virtuelles adaptées à fournir des services optimaux pour divers utilisateurs et applications posent des défis à la gestion efficace des ressources réseau, le traitement des informations sur un volume énorme de trafic, rester robuste face à toutes les menaces potentielles à la sécurité et adapter la fonctionnalité du réseau pour une charge de travail variable en fonction du temps.[1]

Dans ce chapitre nous allons présenter les concepts de l'apprentissage automatique et des réseaux de neurone profond, nous passons par la suite à de rôle de ces derniers a l'automatisation des fonctionnalités de réseau 5G spécialement l'automatisation d'attribution des tranche de réseau 5G. A la fin de ce chapitre nous allons donner une synthèse bibliographique sur les travaux connexes analysés.

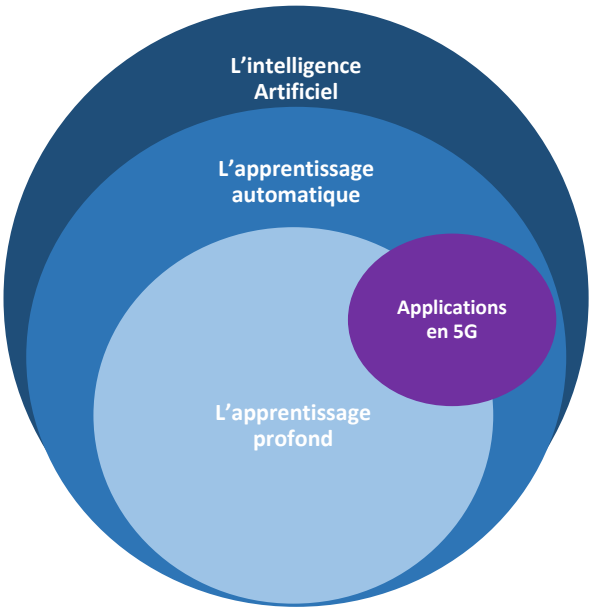


Figure 3.1: Relation entre l'intelligence artificielle, l'apprentissage automatique et l'apprentissage profond. [17]

3.2. Apprentissage automatique

L'apprentissage automatique est une application de l'intelligence artificielle (IA) qui fournit aux systèmes la capacité d'apprendre et d'améliorer automatiquement à partir de l'expérience sans être explicitement programmé. L'apprentissage automatique se concentre sur le développement de programmes informatiques qui peuvent accéder aux données et les utiliser pour apprendre par eux-mêmes. [18]

3.2.1. Les techniques d'apprentissage automatique

Il existe plusieurs types de système d'apprentissage et cela varie en fonction du type de problème que l'on se pose. Il est alors utile de les classer en différentes catégories. Les systèmes de machine Learning peuvent-être classés en fonction de l'importance et de la nature de la supervision qu'ils requièrent durant la phase d'entrainement. On distingue alors quatre grandes catégories : l'apprentissage supervisé, l'apprentissage non supervisé, l'apprentissage semi-supervisé, l'apprentissage avec renforcement.

a) Apprentissage supervisé

L'apprentissage supervisé nécessite une formation avec des données étiquetées qui ont des entrées et des sorties souhaitées et qui nécessite un superviseur qui indique au système le résultat souhaité en fonction des données entrantes. L'utilisation d'algorithmes supervisés permet de prédire, d'estimer ou de classer une variable. L'idée est de former un modèle d'apprentissage qui tente de générer une règle générale mappant les entrées aux sorties avec des échantillons du problème pour lequel la solution est connue. Ensuite, le modèle est utilisé pour trouver des solutions optimales à partir de nouveaux échantillons. [19]

Tout simplement, cet apprentissage représente un ensemble de variable, d'entrée (X : input), une variable de sortie (Y : output), et un algorithme (F : modèle) utilisé pour prédire le résultat Y en se basant sur des donnes X. [20]

D'une manière générale, une machine peut apprendre une relation f : x ➔y qui relie x à y en ayant analysé des millions d'exemples d'associations x ➔y. L'objectif de l'apprentissage supervisé est de développer un modèle (classificateur) capable de classer de nouvelles instances (ne faisant pas partie de X) avec un minimum d'erreur. [20]

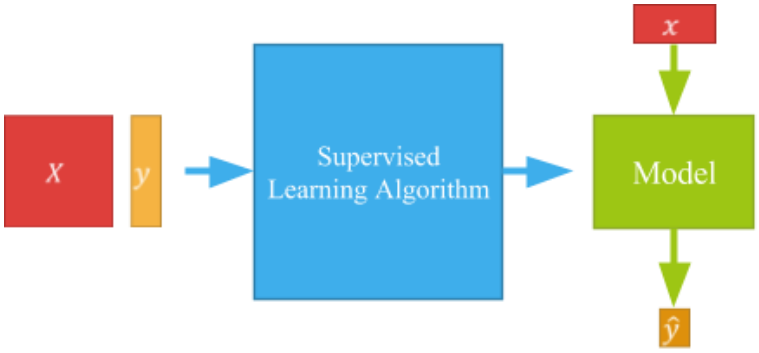


Figure 3.2 : Apprentissage supervisé. [21]

Exemples d'algorithmes d'apprentissage supervisé : [22]

- Réseaux neuronaux.
- Arbre de décision.
- K-voisin le plus proche.
- Régression logistique.
- Machine vectorielle de soutien.
- Bayes naïfs.

Les tâches d'apprentissage supervisées sont divisées en classification et en régression.

- **Classification :** La classification est la tâche de prédire une sortie d'étiquette de classe discrète pour une entrée. [17]

Par exemple donner la taille d'une tumeur comme entré, et le modèle essayé de classer si la tumeur est maligne ou bénigne.

- **La régression :** Il estime la relation entre les variables et prédit la valeur d'une ou de plusieurs cibles à valeur continue**.** [19]

Par exemple prédire une valeur numérique comme le prix d'un appartement, compte tenu d'un ensemble de caractéristiques (emplacement, nombre de chambres, installations).

b) Apprentissage non supervisé

Contrairement à l'apprentissage supervisé, l'apprentissage non supervisé ne nécessite pas de superviseur ; par conséquent, le résultat attendu est inconnu à l'avance (c.-à-d. qu'il n'y a pas de vecteur de sortie). Le but ultime d'apprentissage non supervisé est de trouver des inférences efficientes à partir d'échantillons de données étiquetés pour décrire une

caractéristique ou une structure cachée des données. Il est utile pour identifier une anomalie, reconnaître les modèles ou minimiser la dimensionnalité des données. [19]

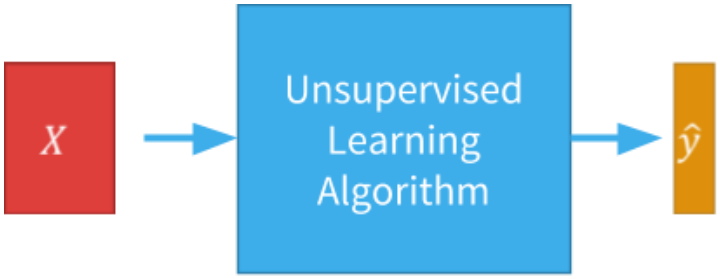


Figure 3.3 : Apprentissage supervisé. [21]

Exemples d'algorithme d'apprentissage non supervisé : [22]

- K-means
- Fuzzy regroupement
- Regroupement hiérarchique.

c) Apprentissage par renforcement

L'apprentissage par renforcement (ou Renforcment learning en anglais) effectue l'apprentissage itératif par une série de renforts par des récompenses ou des punitions. Il apprend à atteindre son but à partir de sa propre expérience. Contrairement à l'apprentissage supervisé, l'apprentissage de renforcement ne le fait pas exigent la fourniture de paires de données d'entrée/sortie correctes et la correction explicite des actions sous-optimales. [1]

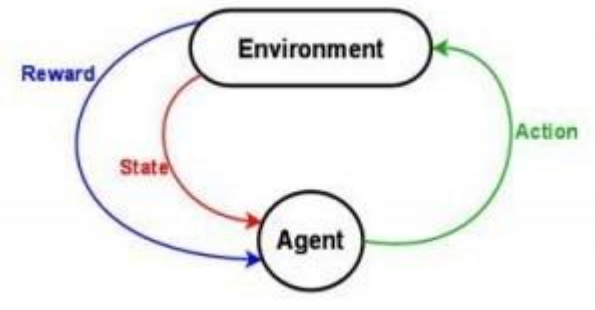


Figure 3.4 : Apprentissage par renforcement. [21]

Exemple d'algorithmes d'apprentissage par renforcement :

- Q-Learning

d) Apprentissage semi-supervisé

L'apprentissage semi-supervisé est un mélange entre l'apprentissage non supervisée et supervisé : où nous commençons par l'apprentissage non supervisé, puis nous poursuivons avec l'apprentissage supervisé. Aussi, l'apprentissage semi supervisée utilise un ensemble de données étiquetées et non- étiquetées. [20]

- Parmi ces trois types d'apprentissage, l'apprentissage supervisé et même semi-supervisé correspond le mieux à la modélisation d'assurance QoS pour les réseaux 5G. [22]

3.3. Classification en apprentissage automatique

La classification est l'un des principaux exemples d'apprentissage supervisé en apprentissage automatique. Compte tenu d'un ensemble de données de formation contenant des observations et leurs résultats catégoriels associés, l'objectif de la classification est d'apprendre une règle générale qui fait correspondre correctement les observations (aussi appelées caractéristiques) aux catégories ciblées. En d'autres termes, un modèle de classification formé sera généré en apprenant à partir des caractéristiques et des cibles des échantillons de formation. Lorsque de nouvelles données ou des données invisibles arrivent, il sera en mesure de déterminer leurs adhésions souhaitées. Les renseignements sur les classes seront prédits en fonction des caractéristiques d'entrée connues à l'aide du modèle de classification formé. [18]

3.3.1. Types de classification

En fonction de la possibilité de sortie de classe, la classification de l'apprentissage automatique peut être catégorisée en classification binaire, classification multi classe et classification multi-étiquettes. [18]

- **La classification binaire :** est le problème de la classification des observations dans l'un des deux possibles des classes. (voir figure 3.5)

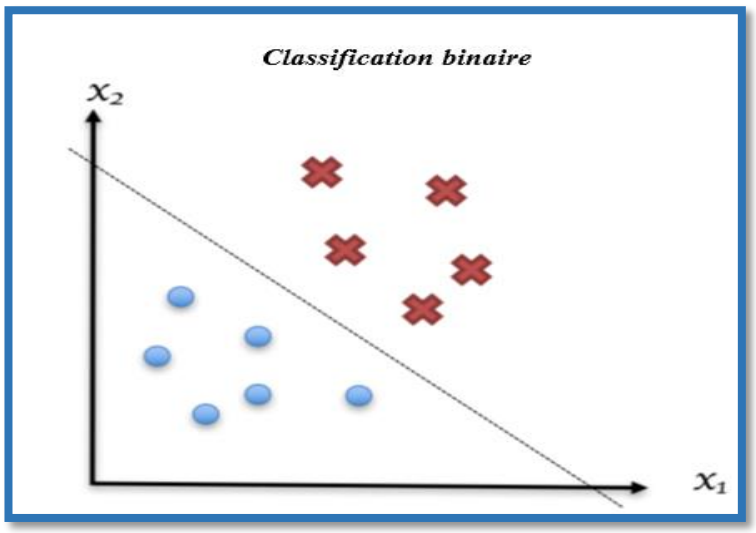


Figure 3.5 : Classification binaire. [18]

- **La classification multi classe (aussi appelée classification multinomiale) :** permet plus de deux classes possibles. (voir figure 3.6)

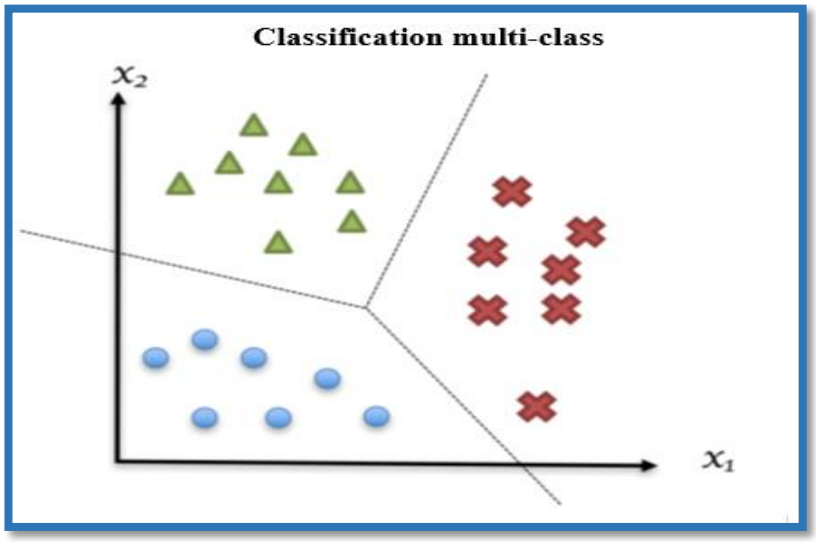


Figure 3.6 : Classification multi-class. [18]

- **La classification multi-étiquettes :** est différente des deux premiers types de classification où la cible et les classes sont disjointes.

3.3.2. Les algorithmes de classification d'apprentissage automatique

a) Séparateurs a vaste marge ou machine a vecteur de support (SVM)

Dans la classification de l'apprentissage automatique, SVM trouve un hyperplan optimal qui sépare au mieux les observations de différentes classes. Un hyperplan est un plan de dimension *n-1* qui sépare l'espace des caractéristiques à *n* dimensions des observations en deux espaces. (voir figure 3.7) [18]

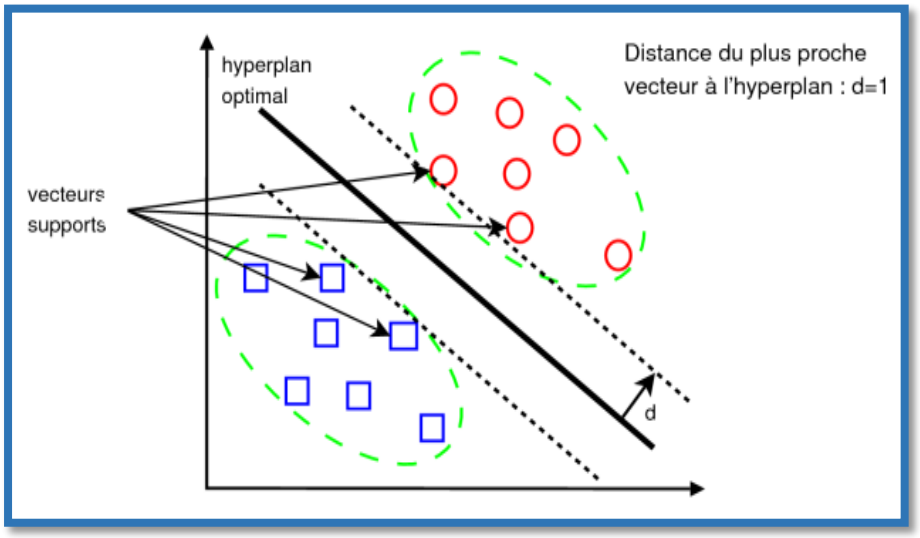


Figure 3.7 : hyperplan optimale et les vecteurs de supports. [18]

b) Forêts d'arbres décisionnels

Forêts aléatoires (en anglais Random Forest - RF) est un algorithme d'apprentissage supervisé. Comme vous pouvez déjà le voir à partir de son nom, il crée une forêt et la rend aléatoire. La «forêt» qu'il construit est un ensemble d'arbres de décision, la plupart du temps formés avec la méthode de «bagging». L'idée générale de la méthode bagging n'est qu'une combinaison de modèles d'apprentissage améliorant le résultat global. [23]

Pour le dire en termes simples ; La forêt aléatoire construit plusieurs arbres de décision et les fusionne pour obtenir une prédiction plus précise et plus stable. Un grand avantage de la forêt aléatoire est qu'elle peut être utilisée pour les problèmes de classification et de régression, qui constituent la majorité des systèmes d'apprentissage automatique actuels. Ci-dessous (figure 3.8), vous pouvez voir à quoi ressemble une forêt aléatoire avec trois arbres

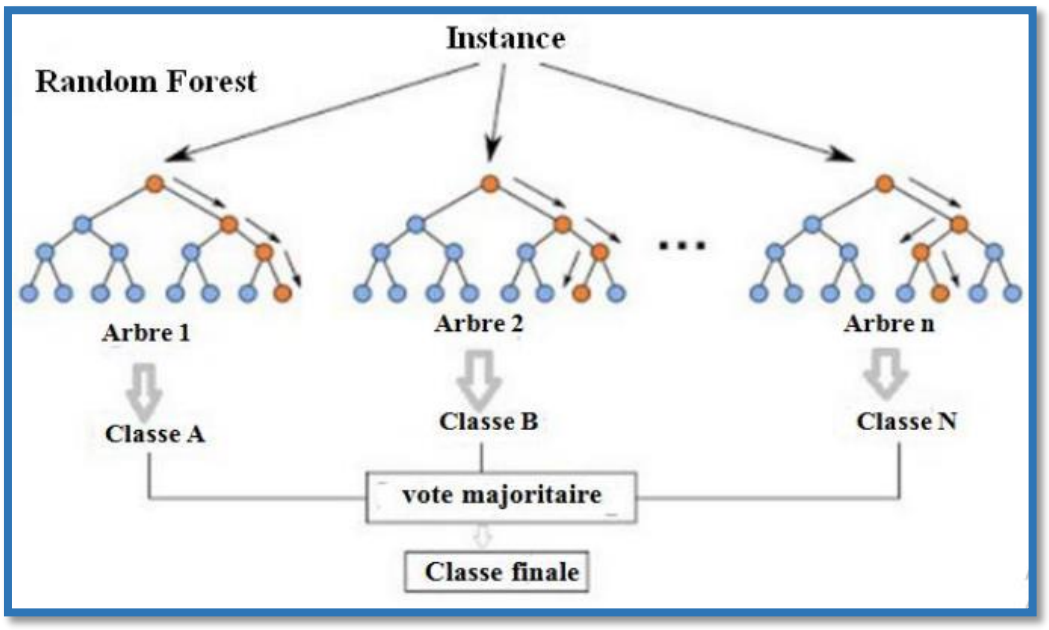


Figure 3.8 : Foret d'arbres décisionnels. [23]

c) Arbre de décision

Les arbres de décision (en anglais décision Tree - DT) sont des arbres qui classifient les instances en les triant en fonction des valeurs des caractéristiques.

Chaque nœud dans un arbre de décision représente une fonctionnalité dans une instance à classer, et chaque branche représente une valeur que le nœud peut prendre.

Les instances sont classées en commençant au nœud racine et triées en fonction de leurs valeurs de fonctionnalité. [24]

d) Bayes naïfs

L'apprentissage Bayésien (en anglais Naïve Bayes-NB) permet de faire des prédictions en se basant sur des probabilités. L'algorithme Naïve Bayes se base sur le théorème de Bayes et suppose l'indépendance totale des variables. C'est un algorithme d'apprentissage supervisé de types classification. [25]

Étant donné les variables de classe, la valeur d'une certaine caractéristique est supposée indépendante de la valeur de toute autre caractéristique par les classificateurs bayésiens naïfs.

La formule bayes est issue des travaux du révérend Tomas Bayes. Elle se base sur la notion de probabilité conditionnelle qui peut se traduire par la probabilité qu'un événement se produise sachant qu'un autre événement s'est déjà produit. [25]

La formule (3.1) est définie par la relation suivante :

$$P(A\,|B\,) = \frac{P(B|A).P(A)}{P(B)} \qquad (3.1)$$

$P(A|B)$ - désigne la probabilité a posteriori de A sachant B,

$P(B|A)$ - désigne la probabilité a posteriori de B sachant A,

$P(A)$ - est la probabilité a priori de A ou probabilité marginale de A,

$P(B)$ - est la probabilité a priori de B ou probabilité marginale de B.

Le but de classificateur Naïve Bayes est de calculer la probabilité conditionnelle :

$$P(C_k|x_1, x_2, x_3, \dots, x_n)$$

Pour chacun des K résultats possibles ou des classesC_k.

Soit$(x = x_1, x_2, x_3, \dots, x_n)$. En utilisant le théorème bayésien, nous pouvons obtenir la formule (3.2) :

$$P(C_k|x) = \frac{P(x\,|C_k).\,P(C_k)}{P(x)} \qquad (3.2)$$

On peut estimer ces paramètres à partir de données (étiquetées), en utilisant le maximum de vraisemblance ou l'estimation MAP Une fois que nous avons appris un classificateur Bayes naïf à partir de données, nous pouvons étiqueter de nouvelles instances en sélectionnant l'étiquette de classe C^* qui a une probabilité postérieure maximale étant donné l'observation $x_1, x_2, x_3, \dots, x_n$.

Sélectionner :$C^*=\text{argmax}_c\ Pr(C|x_1, \dots, x_n)$. [26]

e) *K- Plus Proches Voisins*

L'algorithme des k plus proches voisins (en anglais : K- Nearest Neighbors - KNN*)* est un algorithme d'apprentissage supervisé, il est fondé sur l'analogie suivant : "Dis-moi qui sont tes amis, et je te dirais qui tu es".

Il s'agit d'un algorithme d'apprentissage paresseux non paramétrique, dans lequel la fonction n'est approximée que localement et tous les calculs sont différés jusqu'à l'évaluation de la fonction.

Cet algorithme s'appuie simplement sur la distance entre les vecteurs d'entités et classe les points de données inconnus en trouvant la classe la plus courante parmi les k exemples les plus proches.

La méthode des k-plus proches voisins se base sur une comparaison directe entre le vecteur caractéristique de l'instance à classer et les vecteurs des instances de la base d'apprentissage. La comparaison consiste en un calcul de distances entre ces instances. Puis à classer est assigné la classe majoritaire parmi les classes des k instances les plus proches à cette instance. [27]

- **Notion de distance**

L'algorithme KNN a également besoin d'une fonction de calcul de distance entre deux observations. Plus deux points sont proches l'un de l'autre, plus ils sont similaires et vice versa. Il existe plusieurs fonctions de calcule de distance qui sont choisie en fonction des types de données qu'on manipule. Pour les données quantitatives (exemple : poids, salaires, taille, montant de panier électronique), on a tendance à utiliser la distance euclidienne. Alors que pour des données qui ne sont pas de mêmes types, quantitatives et qualitatives (exemple : âge, sexe, longueur), la distance de Manhattan est plus appropriée. [27]

Dans un hyperespace, espace à n dimensions, la distance euclidienne entre les points. $X(x_1, x_2, x_3, \dots, x_n)$ et $Y(y_1, y_2, y_3, \dots, y_n)$ est donnée par la formule (3.3) :

$$D_e(X, Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \qquad (3.3)$$

La distance de Manhattan entre les point $X(x_1, x_2, x_3, \dots, x_n)$ et $Y(y_1, y_2, y_3, \dots, y_n)$ est donnée par la formule (3.4) :

$$D_m(X|Y) = \sum_{i=1}^{n} |x_i - y_i| \qquad (3.4)$$

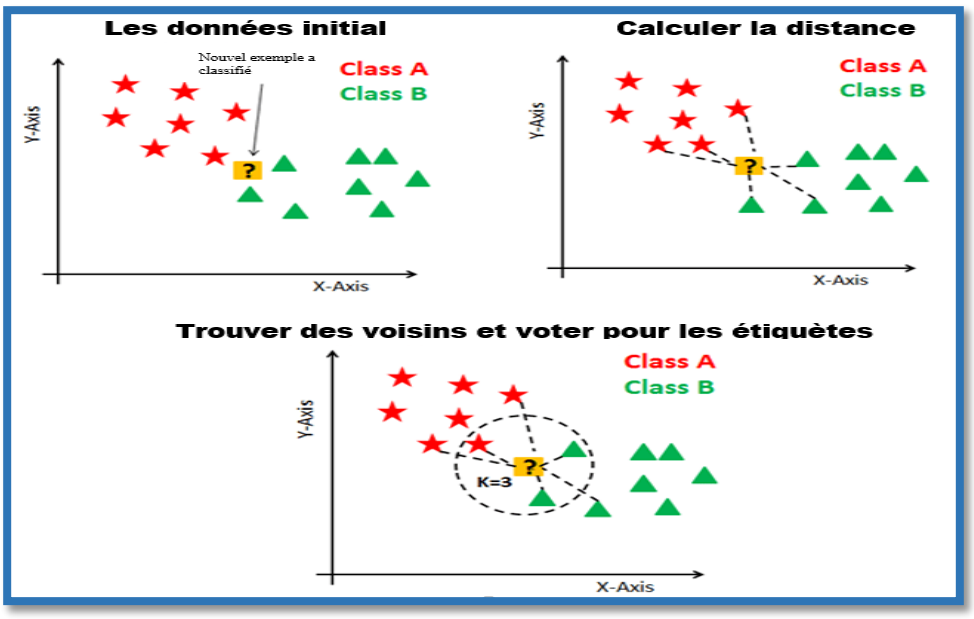


Figure 3.9 : Exemple de K-Plus Proche Voisins. [28]

La figure 3.9 montre une classification avec l'algorithme des k plus proches voisins, ici en prend k = 3. Le nouvel individu sera affecté à la classe Class B (Cardinalité (classe B)> Cardinalité (classe A)).

f) Les réseaux de neurones artificiel

Un réseau neuronal artificiel (en anglais Artificiel Neural Network-ANN) est un aspect de l'IA axé sur l'émulation de l'approche d'apprentissage que les humains utilisent pour acquérir certains types de connaissances. Comme les neurones biologiques, qui sont présents dans le cerveau, ANN contient également un certain nombre de neurones artificiels, et les utilise pour identifier et stocker des informations.[29]

1. Neurones artificiel

Chaque neurone j du réseau est un élément processeur, il est aussi définit comme "une fonction non linéaire, paramétrée, à valeurs bornées". Un neurone reçoit des valeurs en entrée $x_i$associées à des poids w_{ji} représentant l'importance de ces entrées. Il renvoie une valeur unique comme sortie, qui peut être envoyée à plusieurs neurones en aval. Une fonction

de combinaison calcule le potentiel du neurone qui est la somme pondérée des entrées et leurs poids à laquelle se rajoute le seuilb_j: $v_j = b_j + \sum x_i w_{ji}$.

Une seconde fonction ϕ appelée fonction d'activation ou fonction de transfert est appliquée à ce potentiel pour générer la valeur en sortie y_j. [33]

L'association et la connexion de plusieurs neurones selon une architecture donnée permet de construire un réseau de neurones. (voir figure 3.10)

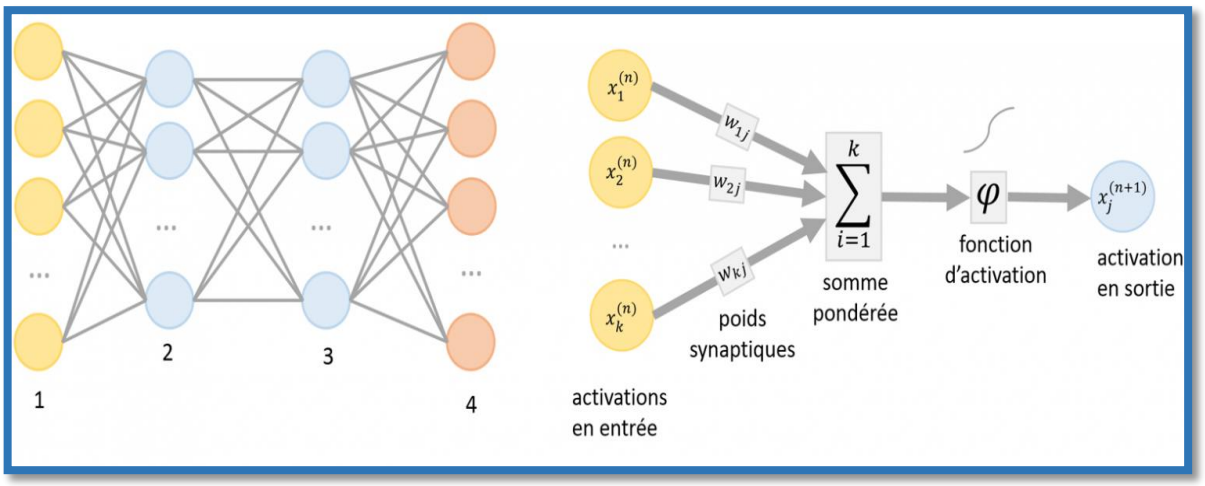


Figure 3.10 : La structure d'un réseau de neurones artificiel. [30]

2. Bloc de construction de base d'un ANN

- **Fonction d'agrégation**

Il existe plusieurs types de fonctions d'agrégation, mais les plus courantes sont : la somme pondérée et le calcul de distance. Le but étant d'associer une seule valeur à l'ensemble des entrées et des poids. [31] [32]

La somme pondérée consiste à calculer la somme de toutes les entrées multipliées par leur poids.

- **Fonction d'activation :**

La fonction d'activation définit le potentiel de sortie d'un neurone en termes de niveau d'activité de ses entrées. Il existe plusieurs fonctions d'activations, les plus utilisées sont représenter dans le tableau suivant (voir tableau 3.1). [32]

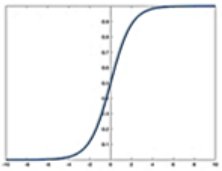
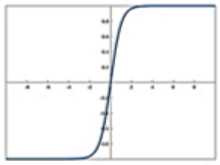
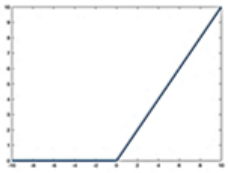
Fonction d'activation	Equation	Graphe
Sigmoid	$S(x) = \frac{1}{1+e^{-x}}$	
Tanh	$\tanh x = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$	
ReLU	$RELU(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x >= 0 \end{cases}$	

Tableau 3.1 : Les différentes fonctions d'activation. [32]

Le choix de la fonction d'activation se révèle dans certains cas être un élément constitutif important des réseaux de neurones.

- **Poids et seuils**

Les neurones se différencient par leurs seuils ainsi que les poids le liant à leurs entrées. Les poids sont accordés à chacune des entrées, permettent de modifier l'importance de certaines par rapport aux autres. Les seuils ou biais quant à eux, permettent d'indiquer quand le neurone doit agir. Il est très difficile de déterminer la valeur des seuils et des poids pour des fonctions complexes. L'apprentissage consiste alors à trouver leurs valeurs optimales afin d'obtenir la sortie voulue. [32]

3. Composition d'un réseau de neurones artificiel

ANN se compose de couches d'entrée et de sortie, ainsi que (dans la plupart des cas) d'une ou de plusieurs couches cachées composées d'unités qui transforment l'entrée en quelque chose que la couche de sortie peut utiliser : [30] [31]

- **Une couche d'entrée :** Elle est constituée de l'ensemble des variables d'entrée.
- **Une couche de sortie :** Elle est constituée de l'ensemble des neurones de sortie du réseau. C'est cette couche-là qui fournit les sorties principales.

- **Une ou plusieurs couches cachées :** Ce sont les couches qui se trouvent entre la couche d'entrée et la couche de sortie. Elles définissent l'activité interne du réseau. En général, les fonctions d'activations sont non linaires au niveau de ces couches.

Les réseaux neuronaux artificiels (ANN) sont divisés en deux catégories : « ANN traditionnels » et « ANN profonds ».

3.4 Apprentissage profond

L'apprentissage profond est un sous-domaine de l'apprentissage automatique, qui est à son tour un sous-domaine de l'intelligence artificielle (IA). Pour une représentation graphique de cette relation, (voir la Figure 3.1) qui exploite de nombreuses couches de traitement non linéaire de l'information pour l'extraction et la transformation de fonctions supervisées ou non supervisées, ainsi que pour l'analyse et la classification des modèles. [34]

L'apprentissage profond est un ensemble d'algorithmes dans l'apprentissage automatique qui tentent d'apprendre à plusieurs niveaux, correspondant à différents niveaux d'abstraction. Il utilise généralement des réseaux neuronaux artificiels. [34]

3.4.1 Le réseau neuronal profond (DNN)

Les DNN sont simplement un ANN avec plusieurs couches cachées entre les couches d'entrée et de sortie et peuvent être supervisées, partiellement supervisées, ou même non supervisées. [39]

Il s'agit d'un domaine de recherche relativement nouveau en apprentissage automatique qui permet aux modèles de calcul composés de multiples couches de traitement d'apprendre des représentations de données complexes à l'aide de multiples niveaux d'abstraction.

3.4.1.1 Les hyperparamètres d'un réseau neuronale profond

Les hyperparamètres sont les variables qui déterminent la structure du réseau (Ex : Nombre d'unités cachées) et les variables qui déterminent la façon dont le réseau est formé (Ex : Taux d'apprentissage). [35]

- **Hyperparamètres liés à la structure du réseau**

1. Nombre de couches et d'unités cachées

✓ Les couches cachées sont les couches situées entre la couche d'entrée et la couche de sortie.

✓ Pour le choix de nombre de ces derniers il suffit de continuer à ajouter des couches jusqu'à ce que l'erreur de test ne s'améliore plus.

✓ De nombreuses unités cachées dans une couche avec des techniques de régularisation peuvent augmenter la précision. Un plus petit nombre d'unités peut entraîner un sous-ajustement.

2. Initialisation des poids du réseau

Idéalement, il peut être préférable d'utiliser différents schémas d'initialisation des poids en fonction de la fonction d'activation utilisée sur chaque couche.

3. Les fonctions d'activation

Sont utilisées pour introduire la non-linéarité dans les modèles, ce qui permet aux modèles d'apprentissage profond d'apprendre des frontières de prédiction non linéaires.

- **Hyperparamètres liés à l'algorithme d'entrainement**

1. Taux d'apprentissage

Le taux d'apprentissage définit la vitesse à laquelle un réseau met à jour ses paramètres.

Un taux d'apprentissage faible ralentit le processus d'apprentissage et un taux d'apprentissage plus élevé accélère l'apprentissage mais peut ne pas converger.

2. Nombre d'époques

Un 'epoch' correspond à un apprentissage sur toutes les données.

3. Taille du lot

La taille du lot est le nombre de sous-échantillons donnés au réseau après lequel la mise à jour des paramètres se produit.

Méthodes utilisées pour déterminer les hyperparamètres :

✓ Recherche manuelle

- ✓ Recherche par grille
- ✓ Recherche aléatoire
- ✓ Optimisation bayésienne

3.5. Défis des réseaux 5G

À l’ère de la 5G, avec la croissance exponentielle du trafic de données réseau et des applications plus riches, on peut facilement prédire que d’énormes quantités de données seront générées dans les réseaux de communication mobiles. Compte tenu de la complexité et de la dynamique croissantes des comportements des réseaux, il est très difficile de planifier les ressources du réseau en fonction des connaissances d’experts, surtout lorsqu’il n’y a pas de relation mathématiquement causale entre les données du réseau et les anomalies QoS. [19]

3.6.L’apprentissage automatique pour le découpage de réseau 5G

Les techniques d'apprentissage automatique sont utiles pour l'automatisation de diverses fonctions des services réseau ; principalement la planification, la conception, l'exploitation, le contrôle, la gestion, la surveillance, la détection des défauts et la sécurité. [36]

Ils fournissent des analyses utiles pour extraire des informations précieuses à partir de données brutes et générer des conseils et des prédictions perspicaces.

L'apprentissage automatique permet à la machine de reconnaître des modèles et des anomalies qu'un humain peut ne pas remarquer ou prendre un temps important.

Ils permettent aussi d'améliorer les performances, de prendre des décisions en raisonnant, de créer et d'exploiter des connaissances, il fait des prédictions et fournit des suggestions sur la base des résultats obtenus en traitant les ensembles de données trop volumineux et trop complexes.

Ainsi, l'adaptation autonome des fonctions du réseau par l'interaction avec les environnements internes et externes peut être réalisée par des techniques d'apprentissage automatique. [1]

a) La planification et la conception

La planification et la conception supposent la prise de décisions fondées sur l'information concernant les exigences de service et le comportement attendu des utilisateurs.

Le processus de conception peut exploiter des techniques d'apprentissage automatique pour l'acquisition de données (extraction de données pertinentes), le traitement de données pour la découverte de connaissances et l'utilisation des connaissances pour le raisonnement et la prise de décisions.

Les techniques d'apprentissage supervisé comme la machine vectorielle de support et l'arbre de décision l'apprentissage non supervisé comme le regroupement spectral peuvent être utilisées pour classer un nouveau service dans l'un des eMBB, mMTC et les catégories uRLLC en fonction des besoins en termes de bande passante, de latence et de taux d'erreur binaire . De même l'algorithme Q-Learning de l'apprentissage par renforcement peut être appliqué pour le raisonnement afin de déterminer les valeurs appropriées des paramètres pour la configuration optimale du réseau.

Cela aiderait à concevoir des tranches de réseau qui seraient adaptées à l'évolution continue de nouveaux services et de nouveaux cas d'utilisation en fournissant la quantité optimale de ressources.[36]

b) Les tâches d'exploitation et de gestion

Les tâches d'exploitation et de gestion visent une utilisation efficace des ressources, tout en satisfaisant de manière optimale les besoins du service et des utilisateurs à tout moment.

Ils doivent comprendre les variations des états du système, connaître les incertitudes, (re) configurer le réseau, prévoir les défis immédiats et proposer en temps opportun des solutions appropriées.

L'allocation et le contrôle des ressources prennent en compte la capacité de calcul du nœud, la bande passante de la liaison, le spectre radioélectrique et l'énergie disponible.

Ils nécessitent des cellules de clustering (pour l'attribution de fréquences et la gestion de l'alimentation), des utilisateurs et des appareils pour une gestion intelligente de la mobilité

ou la mise en place de réseaux de périphérique à périphérique (D2D) pour une utilisation optimale du spectre et de l'énergie.

Les techniques d'apprentissage non supervisé telles que le clustering K-means conviennent à cet effet. [36]

c) La surveillance, la détection des pannes et la sécurité du réseau

La surveillance, la détection des pannes et la sécurité du réseau consistent en des fonctions de collecte d'énormes quantités de données de mesure du système et du rendement, de classification et de regroupement des données pour la contextualisation, la prévision du comportement habituel/inhabituel des utilisateurs et la tendance de l'utilisation des ressources du réseau.

Des algorithmes supervisés tels que la régression logistique peuvent être utilisés pour prédire le comportement inhabituel des appareils ou des utilisateurs en analysant leurs caractéristiques de trafic.

Un DNN aiderait à détecter des menaces sans précédent à la sécurité qui pourraient déboucher sur de nouveaux types de services.

L'analyse du trafic par diverses techniques d'apprentissage non supervisées peut aider à détecter les intrusions et à éviter les attaques par usurpation d'identité.

Les algorithmes classiques d'apprentissage automatique, ils peuvent être modifiés pour améliorer la précision, réduire la complexité, ou leur compromis, et les rendre plus appropriés lors de l'application à l'automatisation du réseau. [36]

3.7.L'apprentissage en profondeur pour le découpage de réseau 5G

Représente un sous ensemble de l'apprentissage automatique, il a été appliqué à des domaines de réseau 5G tels que la classification du trafic, les décisions de routage et la sécurité du réseau.

L'apprentissage profond utilise des réseaux de neurones pour extraire automatiquement l'extraction automatisée des fonctionnalités à partir d'un grands ensembles de données puis utiliser ces fonctionnalités pour classer les entrées, prendre des décisions ou générer de nouvelles informations.

L'apprentissage profondeur offre la possibilité d'identifier et de classer avec précision les applications mobiles, et de créer automatiquement des tranches de réseau adaptatives, entre autres possibilités. [37]

De même, un réseau neuronal profond (DNN) est efficace pour contrôler le trafic réseau hétérogène et atteindre le débit élevé du traitement des paquets et peuvent être aussi appliqués à la sélection des paramètres appropriés pour la reconfiguration du réseau et l'ajustement dynamique des ressources telles que les canaux. [36]

3.8. Travaux connexes

3.8.1. Algorithmes d'allocation des tranches basés sur l'apprentissage automatique dans les réseaux 5G

Gupta et al. dans l'article [38], ont utilisé un algorithme d'apprentissage automatique pour l'allocation des tranches de réseaux 5G. L'état de l'art sur les cas d'utilisation de la 5G a été discuté avec les exigences de QoS (Qualité of Service) telles que : latence, fiabilité, disponibilité et débit.

Ils ont abordé le problème d'allocation de tranches à l'aide d'algorithmes d'apprentissage automatique avec l'ensemble de données public Unicauca-Version-2. La figure 2.11 montre les dix principaux caractéristiques de l'ensemble de données UnicaucaVersion-2 utilisé pour la simulation.

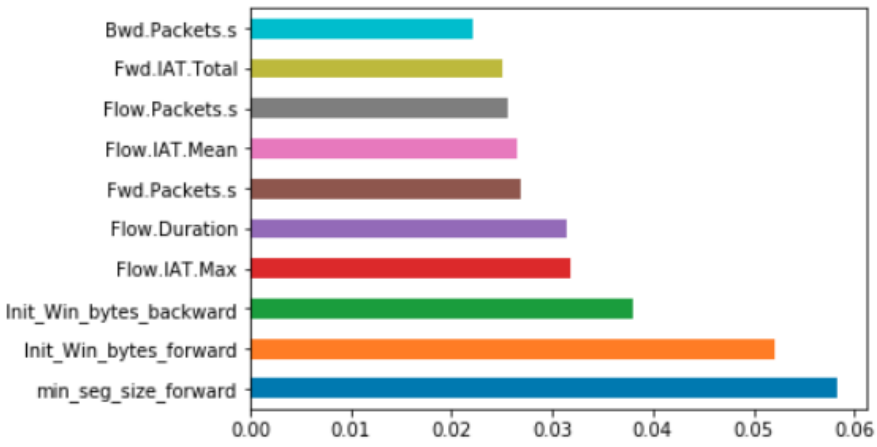


Figure 3.11 : les caractéristiques de Flux de trafic réseau IP utilisées. [38]

Le plus proche voisin (KNN), l'arbre de décision (DT), Foret aléatoire (RF) et machine à vecteur de support (SVM) sont utilisés pour une allocation efficace de tranche pour un service.

Les performances de tous les algorithmes sont testées avec un ensemble de données divisé en modules de formation et de test (voir tableau 3.2).

Split-ratio	Random Forest	SVM	KNN	Decision Tree
90 :10	98.21%	93.04%	99.8978%	94.05%
80 :20	97.96%	92.47%	99.8993%	94.03%
70 :30	97.85%	92.23%	99.8823%	94.13%
60 :40	97.86%	92.23%	99.8823%	94.13%

Tableau 3.2 : Taux d'exactitude d'allocation des slices. [38]

3.8.2. Une approche d'apprentissage profond vers un découpage efficace et fiable du réseau 5G

Thantharate et al. dans l'article [2], ont développé un modèle DeepSlice basé sur le Réseau de neurone d'apprentissage profond (DNN) pour aider à faire la sélection la plus efficace et optimisée des tranches de réseau pour les appareils et/ou les services.

Ils ont utilisé les indicateurs de performance clés(KPIs) du réseau pour entraîner le modèle à analyser le trafic entrant.

3.8.3. Vers l'ultra-latence en utilisant l'apprentissage profond dans le découpage de réseau 5G en appliquant la construction approximative de k-voisin le plus proche graphique (G-KNN)

Gupta et al. dans l'article [39], ont utilisés des modèles de réseaux neuronaux d'apprentissage profond (DNN) pour prédire la tranche appropriée pour le trafic entrant avec des KPIs et des non-KPIs comme métadonnées qui sont construit en utilisant un algorithme de construction de G-KNN pour augmenter l'ensemble de données.

a) Comparaison entre les travaux connexes

Tableau 3.3 montre la comparaison entre les travaux connexes précédentes par rapport les métriques suivant : la techniques d'apprentissage automatique utilisé, le modèles d'apprentissage, dataset et ces caractéristiques utilisée et le taux d'exactitude obtenu par le modèle d'apprentissage appliquée.

Papiers	Technique de ML	Modèle d'apprentissage	Dataset	Caractéristiques	Taux d'exactitude
[38]	Apprentissage automatique supervisé	SVM	Unicauca-Version-2 [40]	Flux de trafic réseau IP (Voir figure 3.11)	Voir tableau 3.2
		RF			
		KNN			
		Decision Tree			
[2]	Apprentissage profond	DNN	DeepSlice dataset [41]	Indicateurs de performances (KPIs)	95%
[39]	Apprentissage profond	DNN	DeepSlice dataset [41]	Indicateurs de performances (KPIs)	68,89%

Tableau 3.3 : Comparaison des travaux connexes.

Après avoir analysé les différents travaux connexes qui traite la même problématique :

- ✓ Les travaux [39] et [2] qui utilisent la même dataset que la nôtre ne passant pas par une étape de prétraitement de données particulièrement l'étape d'équilibrage de données utilisée pour l'entrainement des modèles construit. (voir tableau 3.4)
- ✓ En [38], ils ont utilisé que les quatre algorithmes d'apprentissage automatique motionné.

	Dataset	Approche utilisée	Prétraitement de données
[39]	DeepSlice dataset [41]	DNN	×
[2]	DeepSlice dataset [41]	DNN	×
Notre approche	DeepSlice dataset [41]	ML et DNN	✓

Tableau 3.4 : Les travaux connexes par rapport notre approche

3.9 Conclusion

Dans ce chapitre, nous avons présenté les techniques d’apprentissage automatique en concentrant sur l’approche supervisée spécifiquement la classification.

Ensuite, nous avons présenté l'applicabilité de l'apprentissage automatique pour permettre aux fonctions de découpage 5G d'être exécutées de manière autonome spécifiquement l’attribution de tranche adéquate selon les exigences requises.

A la fin, une présentation avec une synthèse concernant les travaux réalisés pour résoudre le problème d’attribution des tranches de réseau 5G dépendamment des KPIs.

Le chapitre suivant va introduire et discuter nos contributions.

Chapitre 4

Une gestion des tranches du réseau 5G basée sur l'apprentissage automatique

4.1. Introduction

Après avoir étudié les différents techniques d'apprentissage automatique et le réseau de neurones profond, nous allons présenter dans cette partie la méthodologie procédé et ces différent étapes afin d'atteindre notre objectif.

4.2. Objectif

L'objectif de notre travail est de développer un modèle efficace basée sur les techniques d'apprentissage automatique et la technique d'apprentissage profond pour la prédiction du slice approprié pour n'importe quel type d'équipement connecté dépendamment des performances applicatifs requises depuis le trafic entrant .(voir figure 4.1)

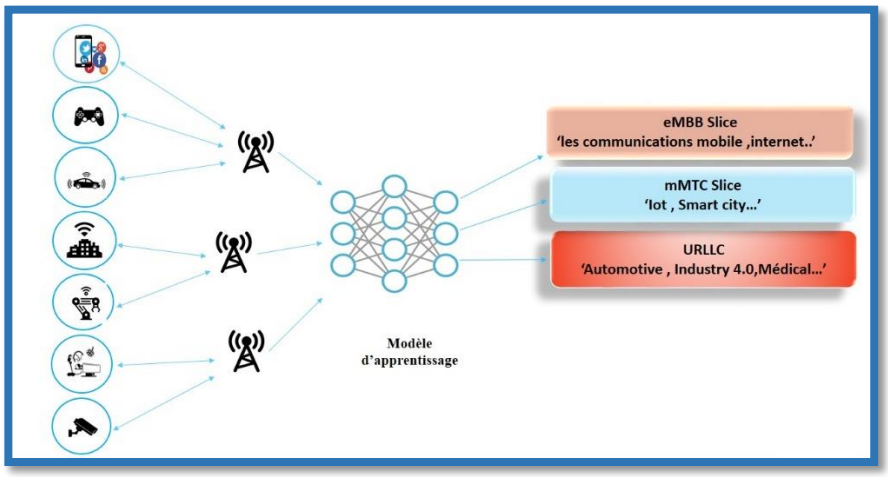


Figure 4.1: Représentation générale de modèle proposé.

4.3. Méthodologie

4.3.1. Processus de modélisation

Notre projet porte sur la classification trafic entrant (dispositif demandant des connexions) selon des tranches du réseau 5G selon avec les méthodes d'apprentissage automatique et d'apprentissage profond. Pour atteindre cet objectif et pour obtenir la meilleure performance possible, La figure 4.2 si dessous décrit les étapes de réalisation de ce projet.

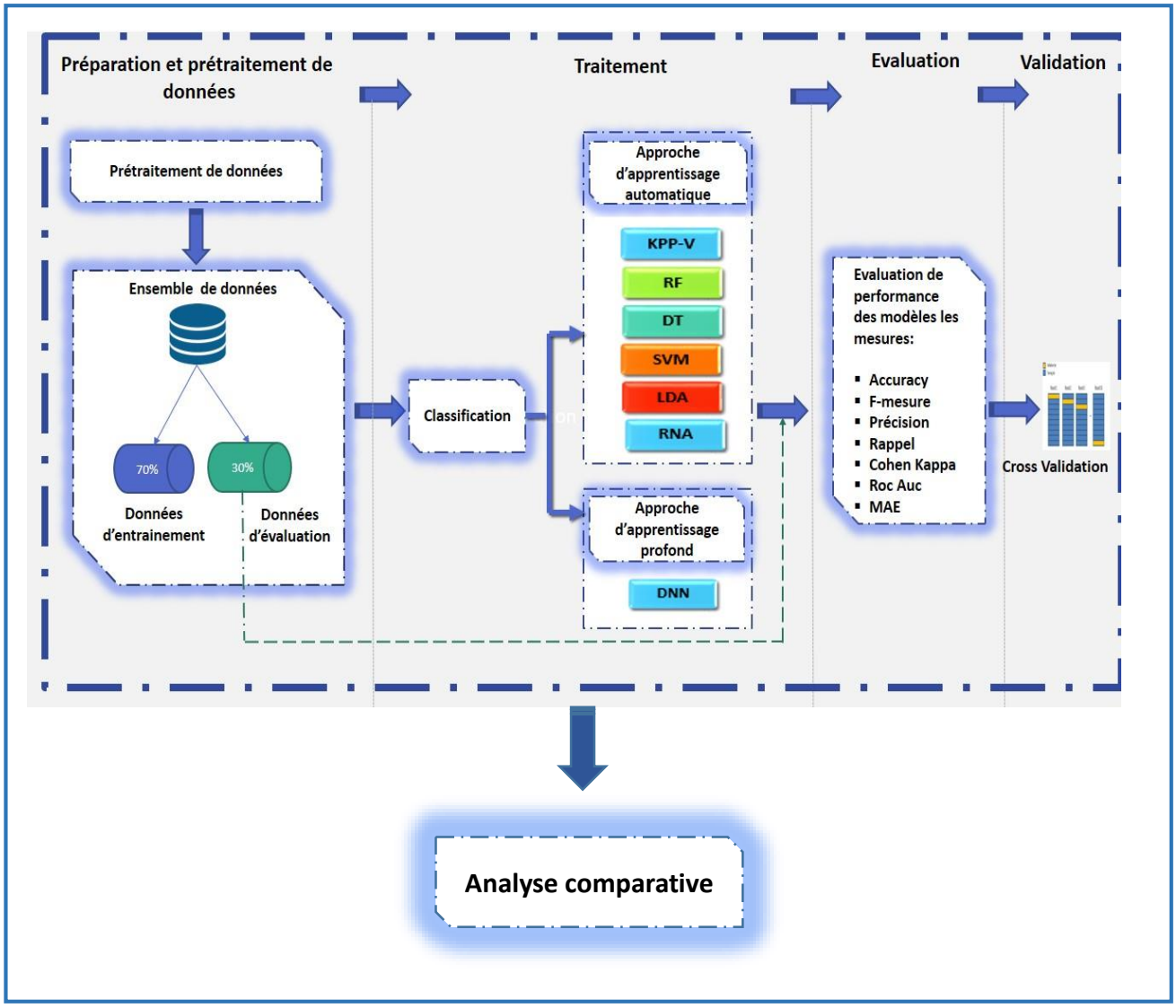


Figure 4.2: Processus de réalisation de projet.

Dans notre projet nous avons suivi les étapes suivant qui sont appliqués pour tous les algorithmes utilisés dans ce projet :

A. Préparation et prétraitements de données.
B. Traitement
C. Evaluation des modèles.
D. Validation des modèles.
E. Etude comparative.

4.3.2. Préparation et prétraitements de données

4.3.2.1. Description de l'ensemble de données

Nous avons principalement étudié la dataset [36], qui contient les KPIs les plus pertinents du réseau et des appareils, tel que le type d'appareil utilisé pour se connecter (téléphone intelligent, appareil IoT, etc.), la catégorie d'équipement utilisateur (UE), l'identificateur de classe QoS (QCI[1]), le budget de retard de paquets, perte maximale de paquets, heure et jour de la semaine, etc. Ces KPIs sont capturés à partir des paquets de contrôle entre l'UE et le réseau. (voir tableau 4.1)

Caractéristique	Description
Type de cas d'utilisation (Use case type)	Type d'appareil utilisé pour se connecter (smartphone, appareil IoT, .. etc.)
Catégorie d'appareil 5G utilisée (5G UE category)	Les informations de catégorie sont utilisées pour permettre à l'gNB de communiquer efficacement avec tous les appareils qui y sont connectés.
Type de technologie supporté (Technology supported)	Par exemple : LTE, 5G…etc.
jour (Day)	Jour de la semaine où la demande de connexion est reçue.
heure (Hour)	Heure de la semaine où la demande de connexion est reçue
Type de ressources (Ressouces types)	Guarented Bit Rate (GBR) Non- Guarented Bit Rate (non-GBR)
Taux de perte de paquet (Packet Error loss rate (PER))	Définit une limite supérieure pour le taux de PDU qui ont été traités par l'expéditeur d'un protocole de couche de liaison, mais qui ne sont pas livrés avec succès par le récepteur correspondant à la couche supérieure
Budget de retard de paquet (Packet Delay Budget (PDB))	Définit une limite supérieure pour le temps pendant lequel un paquet peut être retardé entre l'UE et l'UPF qui termine l'interface N6 (voir architecture 5G chapitre 2 section 2.3).
Tranche de réseau (Slice type)	Fait référence à l'une des tranches

Tableau 4.1: Description des caractéristiques d'ensemble de données.

[1] QCI (Qos Class Identifier) est utilisé dans 3GPP pour identifier un comportement de redirection QoS spécifique pour un flux QoS 5G, il définit le taux de perte de paquets, le budget de retard de paquets, etc.

4.3.2.2. Préparation de l'ensemble de données

Pour notre ensemble de données, nous remarquons le grand écart entre les échantillons des classes, ou on voit sur le graphe en bas (Figure 4.3) comment la classe « emBB » (classe 0) est majoritaire par rapport au les deux autres classes ce qui signifie que **ces données sont un ensemble de données déséquilibrées.**

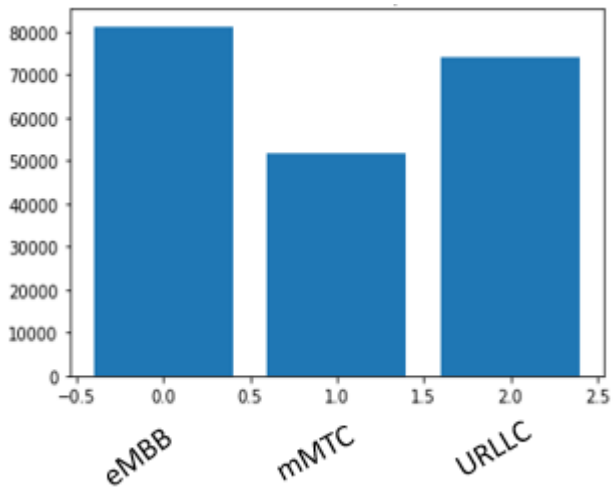


Figure 4.3: Histogramme qui représente déséquilibre d'ensemble de données.

Les données déséquilibrées

Font généralement référence à un problème de classification, où la répartition des classes n'est pas uniforme entre les classes. En général, ils sont composés de deux classes : la classe majoritaire et la classe minoritaire. [42]

- **Techniques de traitement des données déséquilibrées :**

Il y a plusieurs algorithmes qui sont largement utilisés pour traiter la distribution de classe déséquilibrée en cite parmi eux :

1) SMOTE
2) Near Miss Algorithm
3) Random Sampling

4.3.2.3. Prétraitement de données

Le prétraitement des données est un processus de préparation des données brutes et de leur adaptation à un modèle d'apprentissage automatique ou profond.

Les données réelles sont souvent incomplètes, incohérentes et/ou manquantes dans certains comportements ou tendances, et sont susceptibles de contenir de nombreuses erreurs. Le prétraitement des données est une méthode éprouvée pour résoudre ces problèmes.

D'après notre études nous avons résumé les étapes de prétraitement de données en sic étapes présenter dans la figure 4.4.

Figure 4.4: Pipeline de prétraitement des données.

Etape 1 : importation des bibliothèques nécessaires

C'est la première étape du prétraitement des données. Les bibliothèques sont particulièrement utiles pour stocker les routines fréquemment utilisées car nous n'avons pas besoin de les lier explicitement à chaque programme qui les utilise, ce qui nous aide à simplement traiter notre ensemble de données.

Etape 2 : importation d'ensemble de données

Nous pouvons importer l'ensemble de données en plusieurs manière toutes dépendent du format des fichiers de l'ensemble de données tels que (data.xlsx , data.csv …), chaque format d'ensemble de données peut être chargé avec une bibliothèque spécifique. Il

est utile si nous mettons notre ensemble de données dans le même répertoire de travail pour rendre l'accès plus rapide.

Dans notre cas notre ensemble de données est un fichier Excel où Chaque ligne du fichier est un enregistrement de données.

Etape 3 : vérification des valeurs manquantes

Dans le prétraitement des données, il est essentiel d'identifier et de gérer correctement les valeurs manquantes de manière à ne pas réduire les performances de nos modèles d'apprentissage.

Etape 4 : Encodage de données catégorielles

Les modèles d'apprentissage automatique et d'apprentissage profond exigent que toutes les variables d'entrée et de sortie soient numériques. Cela signifie que si nos données contiennent des données catégoriques, nous devons les encoder en nombres avant de pouvoir adapter et évaluer un modèle.

Dans notre ensemble de données les données sont déjà numériques mais nous avons besoin de cette étape car dans notre ensemble de données il y a trois variables de **'Slice Type '** qui sont codées en 0, 1 et 2. Par ces valeurs, le modèle d'apprentissage automatique/profond peut supposer qu'il y a une certaine corrélation entre ces variables qui produiront le mauvais résultat. Donc, pour abordé ce problème, nous allons utiliser l'encodage factice.

- **Variables fictives (ou Dummy variables en anglais) :**

  Les variables fictives sont les variables qui ont des valeurs 0 ou 1

La valeur 1 : désigne la présence de cette variable dans une colonne particulière.

La valeur 0 : désigne l'absence de cette variable dans une colonne particulière.

Etape 5 : Fractionnement de l'ensemble de données

Dans le cas d'apprentissage supervisé, il ne faut pas entrainer et évaluer le modèle sur les mêmes données. Le système doit être évalué sur des données qu'il n'a pas encore rencontrées pour évaluer s'il a bien généralisé à partir des données qu'il a vues déjà. Donc, on à besoin de diviser notre ensemble de données en deux sous-ensembles .il existe plusieurs techniques

de fractionnement de données et dans notre travail nous avons utilisé la technique de Holdout car elle est utile dans le cas où nous avons un grand ensemble de données.

- **Holdout**

C'est quand en divise l'ensemble de données en un ensemble « entraînement » et « évaluation ».

- L'ensemble d'entraînement est ceux sur quoi le modèle est formé.
- l'ensemble d'évaluation est utilisé pour voir dans quelle mesure ce modèle fonctionne sur les données.

Une division commune lors de l'utilisation de la méthode de 'Holdout' consiste à utiliser 70% ou 80% des données pour entrainer le modèle tout en laissant de côté les 30% ou 20% restants.

Dans notre travail, nous avons un grand ensemble de données ; en a environ de 65000 échantillons. Pour construire l'ensemble d'entrainement et d'évaluation, à partir de cet ensemble nous utilisons des ratios 70:30.

Etape 6 : Mise à l'échelle des caractéristiques

La mise à l'échelle des fonctionnalités marque la fin du prétraitement des données.

C'est une méthode pour normaliser les variables indépendantes d'un ensemble de données dans une plage spécifique. En d'autres termes, la mise à l'échelle des caractéristiques limite la plage de variables afin que vous puissiez les comparer pour des motifs communs [43].

Il existe des méthodes pour faire l'étape de mise à l'échelle des caractéristiques en fonction des données telles que la standardisation, la normalisation et la normalisation moyenne.

Il n'y a pas de règle dure et rapide pour dire quand standardiser ou normaliser les données. En commence par l'adaptation de notre modèle aux données brutes, en standardise et normalise et on compare les performances pour de meilleurs résultats.

4.3.3. Traitement

Dans notre projet nous avons choisis six algorithmes d'apprentissage automatique qui sont Random Forest, Decision Tree, LDA, Naïve Bayes, K-NN et un ANN et un algorithme d'apprentissage profond qui est un réseau de neurone profond (DNN) pour la classification de notre ensemble de données. Ce choix est basé sur la nature de notre ensemble de données et la problématique a traité où :

- ✓ Le type de classification dans notre cas est une classification multi classe dés cela en a choisis des algorithmes qui convient avec ce type de classification, d'autre part nous voulons tester différents catégories d'algorithmes pour étudier le meilleur résultat.
- ✓ On a choisis aussi les réseaux de neurone artificiel (ANN) comme une technique de classification et nous avons pris par la suite un réseau plus profond (DNN) afin de le comparé avec d'autres travaux connexes.

A. Modélisation des techniques de classification utilisées

a. Approche d'apprentissage automatique

1) Naïve bayes

Nous classons le trafic entrant au réseau 5G à sa tranche appropriée compte tenu des caractéristiques du paquet Qos.

Les colonnes dans l'ensemble de données représentent ces caractéristiques et les lignes représentent les entrées. Si nous prenons la première ligne de l'ensemble de données, nous pouvons observer (voir tableau 4.3) que la tranche appropriée pour cette demande de connexion est la tranche '0' qui représente la tranche haut débit mobile évolué avec ces KPIs.

Le théorème de Bayes présenté au chapitre 3(équation 3.2) peut être réécrit comme suit :

$$\boldsymbol{P(C_k|kpi_i) = \frac{P(X\,|C_k).P(C_k)}{P(X)}}$$

La variable C_k est la variable de la classe (Slice Type), qui représente la tranche de réseau dans les conditions données.

kpi_1	kpi_2	kpi_3	kpi_4	kpi_5	kpi_6	kpi_7	kpi_8	Slice Type
1	16	1	1	1	2	1	5	0
6	12	2	1	1	2	3	1	2
8	6	2	1	1	1	2	2	2
2	10	2	1	1	2	1	2	1

Tableau 4.2: Extrait de l'ensemble de données

La variable X représente les caractéristiques qui sont des KPIs.

X est donnée comme suit :

$$\boldsymbol{X = \{kpi_1, kpi_2, kpi_3, kpi_4, kpi_5, kpi_6, kpi_7, kpi_8\}}$$

Ici $kpi_1, kpi_2 \ldots, kpi_8$ représentent les caractéristiques et en remplaçant X et en développant en utilisant la règle de chaîne que nous obtenons :

$$\boldsymbol{P(C_k|kpi_1, kpi_2 \ldots, kpi_8) = \frac{P(kpi_1|C_k).P(kpi_2|C_k)\ldots.P(kpi_8|C_k)}{P(kpi_1).P(kpi_2)\ldots.P(kpi_8)}} \quad \textbf{(4.1)}$$

Le classificateur bayésien va choisir la classe « C_k » qui a la plus grande probabilité, on parle de règle MAP (maximum a posteriori)

2) Réseau de neurone artificiel (ANN)

Le réseau de neurone reçoit les entrées qui sont les KPIs $\{\boldsymbol{kpi_1, .., kpi_8}\}$ afin qu'il puisse déterminer une sortie qui représente l'une des tranches de réseau (classe 0, 1,2) appropriée (voir figure 4.5) en peut décrire le processus de passage de l'entrée jusqu'à l'arrivée à la sortie comme suit :

- Les entrées (kpi) reçues de la couche d'entrée sont multipliées par le poids attribué w.
- Les valeurs multipliées sont ensuite ajoutées pour former la somme pondérée.
- La somme pondérée des entrées et de leurs poids respectifs est ensuite appliquée à une fonction d'activation pertinente.
- La fonction d'activation fait correspondre l'entrée à la sortie correspondante.

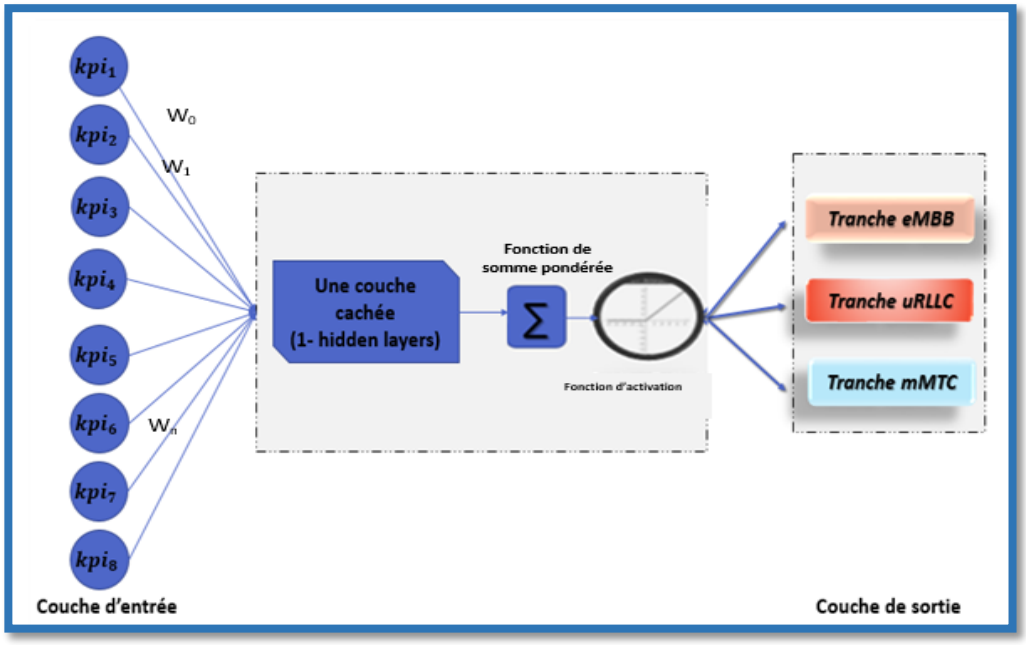


Figure 4.5: Modélisation de modèle d'apprentissage automatique Réseau de neurone artificiel.

3) Foret d'arbres décisionnels (RF)

La Figure 4.6 illustre la modélisation d'apprentissage automatique typique avec des arbres décisionnels et prédire le résultat avec le vote majoritaire. Conformément à notre ensemble de données d'entrées, nous avons environ 8 entrées différentes qui contribueront ensemble à une décision que le modèle prendra.Lors de l'entrainement de nos données, RF construit plusieurs arbres décisionnels basés sur les entrées, chaque branche d'un arbre représente une occurrence ou une réponse possible.

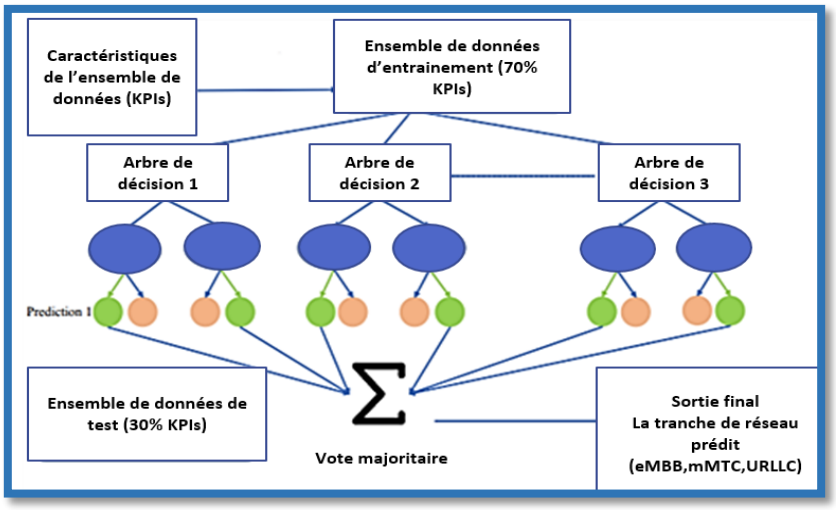


Figure 4.6: Modélisation de modèle d'apprentissage automatique Random Forest.

4) Arbre de décision (DT)

L'algorithme d'Arbre de décision prend comme entrée l'ensemble d'entrainement de données qui sont les KPIs et prédit à la fin la valeur de sortie qui constitue l'une des tranches de réseau 5G.(voir figure 4.7)

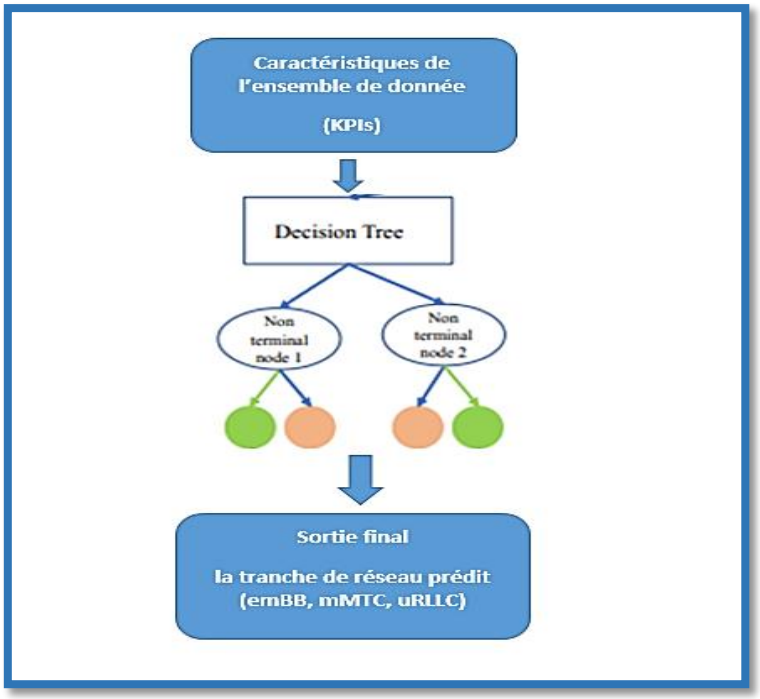


Figure 4.7: Modélisation de modèle d'apprentissage automatique arbre de décision (DT).

5) K-Plus Proche Voisins (KNN)

K-plus proche voisins est l'un des algorithmes les plus faciles à la mise en œuvre.

On suppose l'ensemble d'entrainement D et un objet de test x_{new} = (x', C'), l'algorithme calcule les distances entre x_{new} et tout l'objet d'entraînement (x, C) qui appartient à D en utilisant la fonction euclidienne pour déterminer sa liste des voisins les plus proches (x représente la donnée d'entrainement et C sa classe approprié, x' représente la donnée d'évaluation et C' sa classe approprié).(voir figure 4.8)

La fonction de calcule de distance peut être écrite comme suit :

$$D\,(x, x_{new}) = \sqrt{\sum_{i=1}^{n}(x_i - x_{new_i})^2} \qquad (4.1)$$

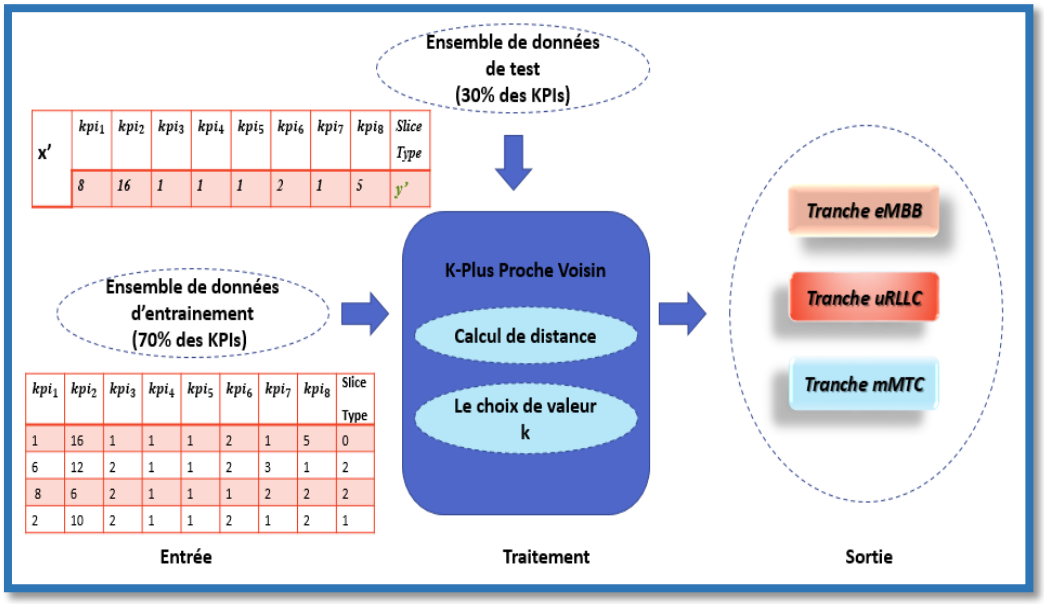


Figure 4.8: Modélisation de modèle d'apprentissage automatique K-Plus Proche Voisin (K-NN).

6) Analyse discriminante linéaire (en anglais linear discriminant analysis ou LDA)

LDA prend comme entrée l'ensemble de données (KPIs) et sert à la séparation des données a fin de déterminer la sortie appropriée. (voir figure 4.9)

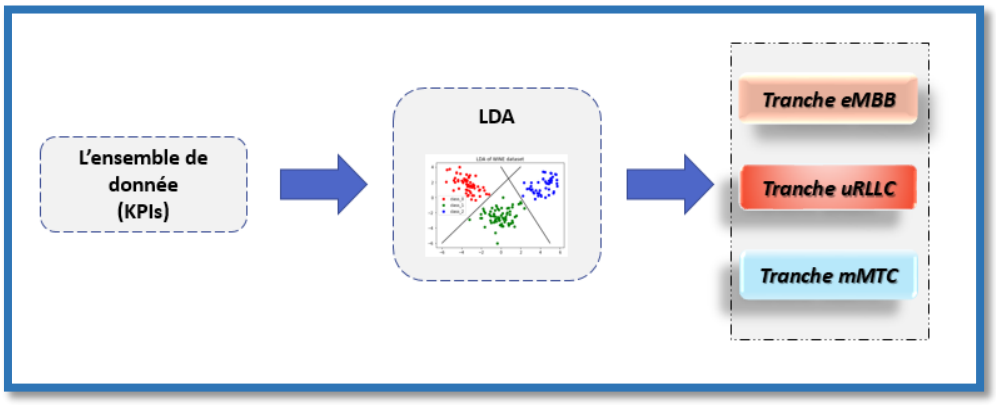


Figure 4.9: Modélisation de modèle d'apprentissage automatique analyse discriminante linéaire.

b. Approche d'apprentissage profond

1) Réseau de neurones profond

Le réseau de neurones profond (DNN) fonctionne bien avec un bon taux d'exactitude lorsque les données sont volumineuses et non structurées.

Nous utilisons le DNN avec l'ensemble de données des KPIs afin qu'il puisse prédire une tranche pour un dispositif à partir de trafic entrant.

Le modèle DNN apprend le type de périphérique attribué à quelle tranche et prédit avec précision pour les futures demandes de connexion au réseau.

Dans la Figure 4.10, nous montrons l'aperçu du notre modèle DNN.

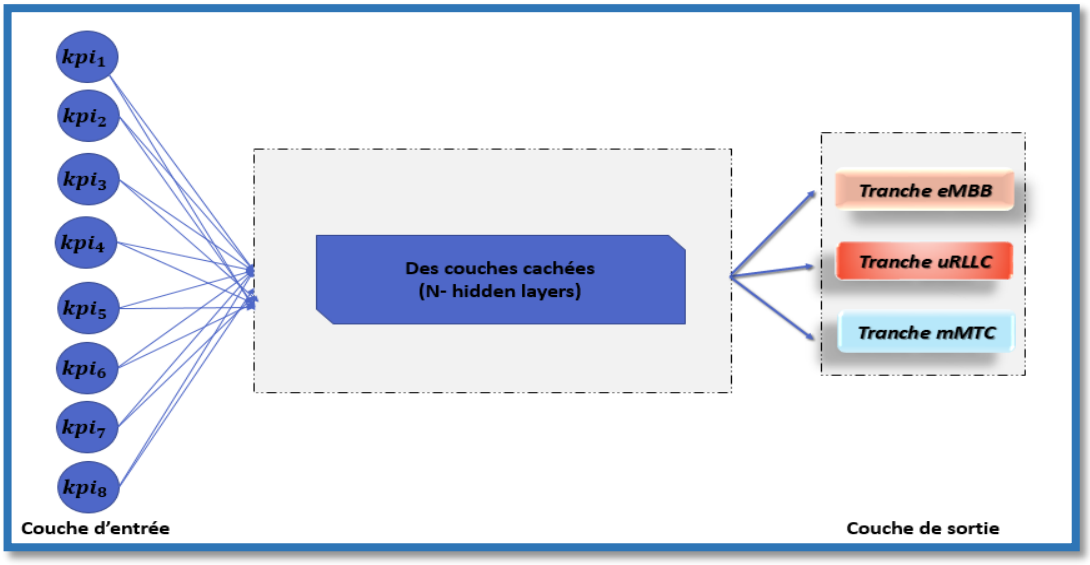


Figure 4.10: Modélisation de modèle d'apprentissage profond de réseau de neurone profond.

4.3.4. Evaluation des performances

Après avoir appliqué des algorithmes d'apprentissage automatique et profond, nous avons besoin quelques outils pour savoir dans quelle mesure ils ont accompli leur travail.

Ces outils sont appelés métriques d'évaluation des performances. Un nombre important de paramètres ont été introduits dans les études, où chacun considère certains aspects d'une performance de l'algorithme. Ainsi, pour chaque problème d'apprentissage automatique ou profond, nous avons besoin d'un ensemble approprié de mesures pour l'évaluation des performances. [44]

Dans notre travail, nous utilisons plusieurs métriques communes pour les problèmes de classification pour obtenir des informations précieuses sur les performances des algorithmes choisis et pour faire une analyse comparative est c'est le but de ce travail.

A. Les métriques de mesure de la performance des modèles de classification

1. La matrice de confusion (ou confusion matrix en anglais)

Une matrice de confusion est une sorte de tableau qui définit le nombre d'instances de données qui sont mal classées et qui sont correctement classifiées. Il s'agit d'une matrice n×n où « n » est le nombre de classes définies pour l'ensemble de données (dans notre projet n=3 car en a trois classe 0,1 et 2).

La figure 4.11 présente un exemple de matrice de confusion et montre les valeurs des vraie positive, vraie négative, faux positive et des faux négative pour la classe 1.

Cette matrice de confusion sert de base pour calculer les valeurs pour presque tous les autres paramètres utilisés (rappel, précision…etc.). Les colonnes représentent les valeurs prédites par le classificateur et les lignes représentent les valeurs réelles ou les étiquettes de classe auxquelles l'objet de données appartient réellement. Les valeurs dans les cellules sont :

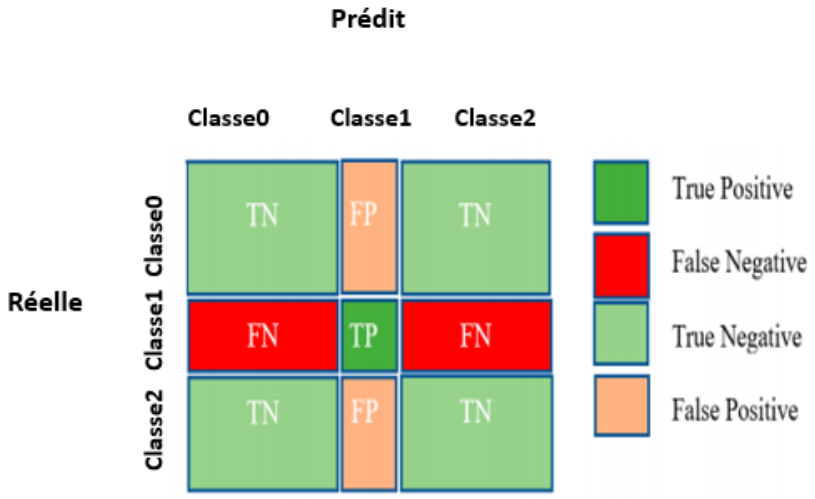


Figure 4.11: Exemple de matrice de confusion pour une classification multi classe. [44]

- **Vrai négatif (VN) :** les cas où la prédiction est négative, et où la valeur réelle est effectivement négative.
- **Faux positif (FP) :** les cas où la prédiction est positive, mais la valeur réelle est négative. Exemple : la valeur réelle appartient à une classe 0 (haut débit mobile évolué- eMBB) mais elle n'est pas prédite qu'elle appartient à cette classe. Calculé comme suit :

- **Faux négatif (FN) :** les cas où la prédiction est négative, mais où la valeur réelle est positive. Exemple : le trafic entrant au réseau appartient à une classe 0 (haut débit mobile améliorer- eMBB) et il est affecté vers une autre tranche.
- **Vrai positif (VP) :** les cas où la prédiction est positive, et où la valeur réelle est effectivement positive et correctement classés .C'est le cas où chaque trafic entrant est attribué à la tranche appropriés correctement.

2. Le Taux d'exactitude, la précision,spécificité et le rappel

1) **La Précision :** Il montre simplement «quel nombre d'éléments de données sélectionnés sont pertinents». En d'autres termes, parmi les observations qu'un algorithme a prédites comme positives, combien d'entre elles sont réellement positives.

$$précision = \frac{VP}{VP + FP} \qquad (4.2)$$

2) **Le Rappel :** Appelé aussi sensitivité c'est le pourcentage des instances positives correctement identifiées. [44]

$$rappel = \frac{VP}{VP + FN} \qquad (4.3)$$

3) **Spécificité :** Elle indique quelle fraction de tous les échantillons négatifs est correctement prédite comme négative par le classificateur.

$$Spécificité = \frac{VP}{VP + FP} \qquad (4.4)$$

4) **La mesure F (ou F-measure en anglais) :** Cette mesure tient compte de la précision et du rappel pour calculer la performance d'un algorithme. Mathématiquement, c'est la moyenne harmonique de précision et de rappel formulée comme suit :

$$\mathrm{F-measure} = 2 \times \frac{précision \times rappel}{précision + rappel} \qquad (4.5)$$

5) **Le taux d'exactitude (ou accuracy en anglais) :** C'est le plus utilisé et peut-être le premier choix pour évaluer la performance d'un algorithme dans les problèmes de classification. Il peut être défini comme le rapport entre les éléments de données classifiés avec précision et le nombre total d'observations. Malgré la facilité d'utilisation

généralisée, le taux d'exactitude n'est pas la mesure de rendement la plus appropriée dans certaines situations, surtout dans les cas où les classes de variables cibles de l'ensemble de données sont déséquilibrées.

$$Taux\ d'exactitude = \frac{VP + VN}{VP + VN + FP + FN} \qquad (4.6)$$

2. Autres métriques

5) L'erreur absolue moyenne (EAM ou MAE pour Mean Absolute Error) : Désigne la différence entre la prédiction d'une observation et la valeur réelle de cette observation.

6) La statistique de Cohen Kappa : est une autre métrique qui est utile surtout dans les cas où les données ne sont pas équilibrées ou pour des cas de classification multiple, est la statistique de Cohen Kappa. Cette statistique mesure l'accord entre deux classificateurs qui classent chacun des n éléments dans C classes mutuellement exclusives. Elle inclut la distribution marginale de la variable cible dans le calcul de la performance.

7) L'aire sous la courbe ROC (AUC : Area Under the Curve) : Représente une mesure qui permet de quantifier numériquement la performance de nos classificateurs. L'AUC est très utile quand on veut faire une comparaison entre différents modèles et c'est notre cas. Le meilleur modèle est celui qui a l'AUC le plus élevé.

4.3.5. La validation

Après avoir appris quelles mesures sont utilisées pour mesurer un modèle de classification, nous pouvons maintenant étudier comment le mesurer correctement. Nous ne pouvons tout simplement pas adopter les résultats de la classification à partir d'un ensemble de tests fixes, au lieu de cela, la technique de validation croisée k-fold est utilisée pour évaluer comment un modèle fonctionnera généralement dans la pratique.[45]

- **La validation croisée (ou Cross validation en anglais) :** est l'une des techniques utilisées pour tester l'efficacité d'un modèle, elle comprend les étapes suivantes (voir figure 4.12) :

1. Divise les n observations de l'ensemble de données en k sous-ensembles de taille mutuellement exclusifs et égaux ou presque égaux appelés « plis ».
2. Ajustez le modèle en utilisant les plis k-1 comme ensemble d'entraînement et un pli (k ième) comme ensemble de test. Une fois chaque itération terminée, enregistrez l'erreur du modèle.
3. Répétez ce processus k fois en utilisant un pli différent à chaque fois comme ensemble de test et les plis restants (k-1) comme ensemble d'apprentissage.
4. Une fois toutes les itérations terminées, prenez la moyenne des k modèles. Ce serait l'erreur quadratique moyenne du modèle.

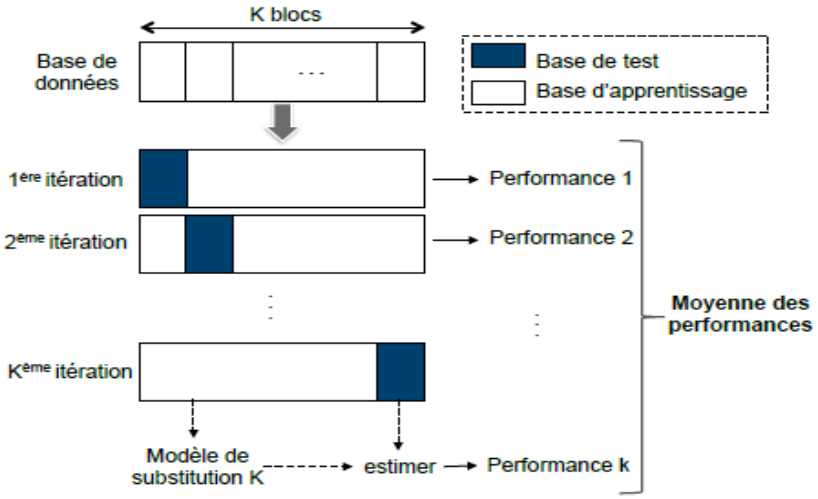


Figure 4.12: Méthode de validation croisée K-Fold. [45]

4.4. Conclusion

Dans ce chapitre nous avons présenté l'architecture générale et détaillé dont le quelle on mettre les différent étapes du projet, en commençant par l'analyse de données, les modèles utilisées ainsi que les techniques d'évaluations.

Dans ce qui suit, nous allons détailler les environnements et les outils utilisés pour réaliser la méthodologie proposée et discuter les résultats obtenus.

Chapitre 5

Résultats expérimentaux et discussions

5.1. Introduction

Après avoir décrit notre solution de façon conceptuelle dans le chapitre 4, dans ce chapitre nous allons présenter les différentes résultats expérimentaux .On commence par la description de l'ensemble de données utilisée, nous présentons par la suite les outils utilisés pour la mise en œuvre de notre projet, ensuite, nous présentons les résultats obtenus et évaluons les performances des méthodes de classification utilisées à travers les métriques d'évaluation choisies dans la phase conception. Enfin, nous allons présenter une étude comparative entre les différentes méthodes utilisées.

5.2.Environnements et outils de développement utilisés

Pour réaliser les étapes conçue, différents environnements et outils sont utilisés (voir tableau 5.1).

	Logo d'outils	Description
Outils commun	Google Colaboratory	**Google Colab ou Colaboratory** est un service cloud, offert par Google (gratuit), basé sur Jupyter Notebook et destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning et Deep Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur notre ordinateur à l'exception d'un navigateur.
	jupyter	**Jupyter Notebook** est une application Web Open Source permettant de créer et de partager des documents contenant du code (exécutable directement dans le document), des équations, des images et du texte. Avec cette application il est possible de faire du traitement de données, de la modélisation statistique, de la visualisation de données, du Machine Learning, etc.
	python	**Python** est un langage de programmation interprété, orienté objet, de haut niveau avec une sémantique dynamique,.Il est très sollicité par une large communauté de développeurs et de programmeurs. Les packages (bibliothèques) de python encouragent la modularité et la réutilisabilité des codes. Il est utilisé pour le développement web, l'IA, l'apprentissage automatique, les systèmes d'exploitation, le développement d'applications mobiles, les jeux vidéo et bien d'autres.

	NumPy	**NumPy** est une extension du langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.
	matplotlib	**Matplotlib** est une bibliothèque Python open source permettant de créer des visualisations de données.
	pandas	**Pandas** est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques.
Outils pour l'apprentissage profond	Keras	**Keras** est une API de réseau de neurones écrite en langage Python. Il s'agit d'une bibliothèque Open Source, exécutée par-dessus des frameworks tels que Theano et TensorFlow. Conçue pour être modulaire, rapide et simple d'utilisation, Keras a été créée par l'ingénieur François Chollet de Google. Elle offre une façon simple et intuitive de créer des modèles de Deep Learning.
Outils pour l'apprentissage automatique	scikit learn	**Scikit-learn** est une bibliothèque libre python destinée à l'apprentissage automatique. Elle comprend notamment des fonctions pour estimer des algorithmes de classification. Elle est conçue pour s'harmoniser avec d'autres bibliothèques libres python, notamment NumPy.

Tableau 5.1: Outils de dévlopement .

5.3. Description d'Environnement

Afin d'entraîner notre ensemble de données, nous avons besoin d'un calcul parallèle avec un minimum de ressources pour gagner du temps et entraîner les différents modèles proposées .Pour cela nous avons utilisé Google Colab le service de cloud gratuit fournit par Google qui prend en charge le GPU libre, et elle nous offre un seul GPU NVIDIA Tesla K80 de 12 Go qui peut être utilisé jusqu'à 12 heures en continu. (voir figure 5.1)

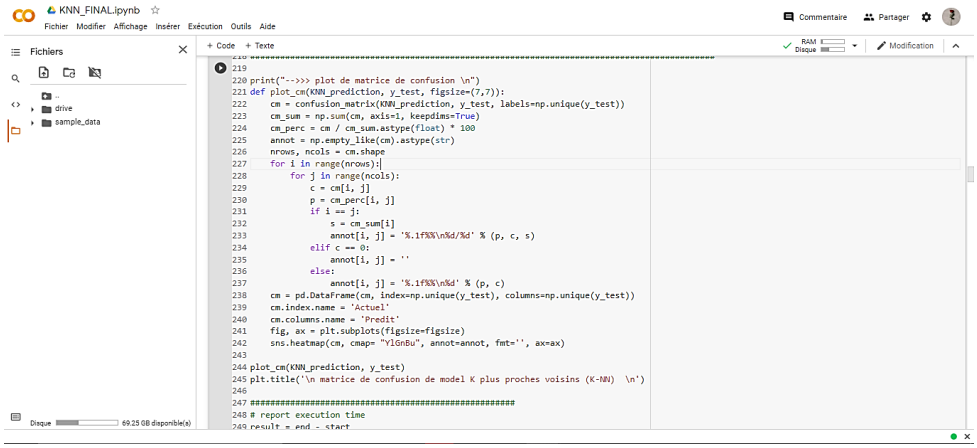


Figure 5.1: Environement de travail

5.4. Analyse de dataset

5.4.1. Aperçu sur de l'ensemble de données

Les caractéristiques de l'ensemble de données utilisées sont résumées dans le tableau si dessous (voir tableau 5.2).

Nombre des caractéristiques (features)	8
Nombre d'échantillons	207000
Format	Excel

Tableau 5.2: Description d'ensemble de données.

L'ensemble de données est constitué d'un ensemble d'environ de 207 000 lignes et 9 colonnes.

- les colonnes de 1 jusqu'à 8 sont des attributs (features).
- La 9éme colonne représente la classe cible (label) de l'ensemble de données qui peut être soit 0,1 et 2, qui représente respectivement eMBB, mMTC et uRLLC.

	A	B	C	D	E	F	G	H	I
1	Use CaseType (Input 1)	LTE/5G UE Category (Input 2)	Technology Supported (Input 3)	Day (Input4)	Time (Input 5)	GBR (Input 6)	Packet Loss Rate (input 7)	Packet Delay Budget (input8)	Slice Type (Output)
2	1	1	1	1	0	2	1	5	0
3	1	1	1	1	1	2	1	5	0
4	1	1	1	1	2	2	1	5	0
5	1	1	1	1	3	2	1	5	0
6	1	1	1	1	4	2	1	5	0
7	1	1	1	1	5	2	1	5	0
8	1	1	1	1	6	2	1	5	0
9	1	1	1	1	7	2	1	5	0
10	1	1	1	1	8	2	1	5	0
11	1	1	1	1	9	2	1	5	0
12	1	1	1	1	10	2	1	5	0
13	1	1	1	1	11	2	1	5	0
14	1	1	1	1	12	2	1	5	0
15	1	1	1	1	13	2	1	5	0
16	1	1	1	1	14	2	1	5	0
17	1	1	1	1	15	2	1	5	0
18	1	1	1	1	16	2	1	5	0
19	1	1	1	1	17	2	1	5	0
20	1	1	1	1	18	2	1	5	0
21	1	1	1	1	19	2	1	5	0

Figure 5.2: Ensemble de donnée des KPIs.

5.4.1. Prétraitement de l'ensemble de donnée

a) Importer les bibliothèques

Tout d'abord, on commence par importer les différentes bibliothèques et modules nécessaires pour travailler avec les données tels qu'ils sont présentés dans les figures ci-dessous.

- **Les bibliothèques de prétraitement et visualisation de données**

```
########### biblio plot ##########################################"
import matplotlib.pyplot as plt
from matplotlib import pyplot
from itertools import cycle
from scipy import interp
import seaborn as sns
############## preprétraitement d'ensemble de données ##############
from sklearn.preprocessing import LabelEncoder
from collections import Counter
from imblearn.over_sampling import SMOTE #balancer dataset
####################################################################
import  pandas  as pd
from time import time
from matplotlib import pyplot
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold
####################################################################
```

Figure 5.3: Bibliotheques utilisée pour la visualisation et prétraitement de données.

- **Les bibliothèques utilisées pour l'évaluation des modèles**

```
############# biblio d'évaluation ################################
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_curve, auc
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import cohen_kappa_score
from sklearn.metrics import roc_auc_score
from sklearn.metrics import precision_recall_curve
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import StandardScaler
```

Figure 5.4: Bibliothèques utilisées pour l'évaluation des modèles.

- **Les bibliothèques communes pour les modèles de l'apprentissage automatique**

```
############ biblio pour les modéles d'apprentissage automatique #########
from sklearn.linear_model import MLP
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier
```

Figure 5.5: Les bibliothèques utilisées pour la construction des modèles d'apprentissage automatique.

- **Les bibliothèques utilisées pour l'apprentissage profond**

Pour la construction d'un réseau de neurone avec Keras, nous devons importer les différentes bibliothèques et modules tels qu'ils sont présentés dans la figure 5.5

```
""" biblio keras"""
from keras import Sequential
from keras.layers import Dense
from keras.utils import np_utils
```

Figure 5.6: Les bibliothèques utilisées pour la construction de modèle d'apprentissage profond.

b) Importer l'ensemble de données

Nous commençant par l'importation de l'ensemble de données, nous avons utilisée **read_excel ( )** de la bibliothèque pandas puisque l'ensemble de données utilisé est de format Excel.

```
47 dataset = pd.read_excel('/content/drive/MyDrive/Colab Notebooks/data_kh.xlsx')
```

Figure 5.7: importation de l'ensemble de données.

- **Séparer les caractéristiques et les étiquette**

- les caractéristiques : les caractéristiques sont définies comme des variables indépendantes individuelles qui agissent en tant qu'entrée. Les modèles de prédiction utilisent les caractéristiques pour effectuer des prédictions.

- Les étiquettes : les labels sont la sortie finale, c'est-à-dire le résultat de la prédiction.

Dans notre cas les caractéristiques prennent les colonnes de 1 à 8 et les labels la colonne 9.

([: ,après :avant] et [:,aprés])

```
50 # Séparation entre les caractéstiques et les étiquettes
51 X = dataset.iloc[:,0:8].values
52 Y = dataset.iloc[:,8].values
```

Figure 5.8: Séparation entre les caractéristiques et les étiquettes.

c) Visualiser l'ensemble de données

L'étape de visualisation des données est une étape très importante et qui sert à mieux comprendre les données à utiliser.

On a visualiser aussi la distribution de différentes classes 0,1 et 2 qui représente respectivement les tranches emBB, mMTC et uRLLC (Voir figure 5.9)

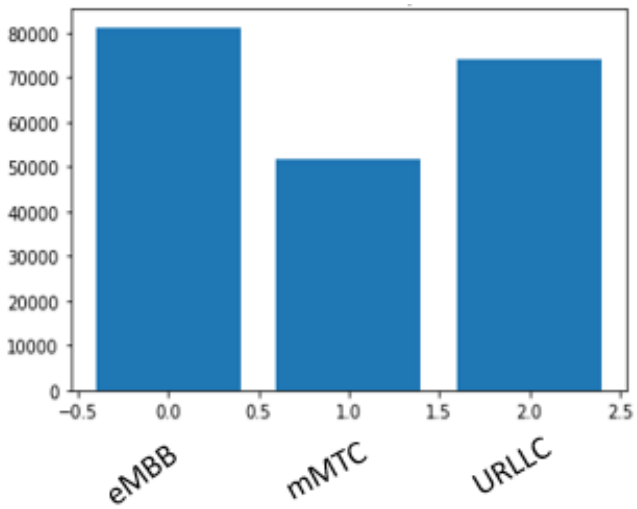


Figure 5.9: Distributions de classe de l'ensemble de données.

d) Vérification des valeurs manquantes

Comme nous avons cité dans chapitre 3 (section 4.3.1.3) la vérification des valeurs manquantes est une étape importante car elle influe sur la performance de nos modèles, nous avons utilisé la fonction **isnull()** pour effectuer cette étape.

Toutes les valeurs de notre ensemble de données ne sont pas nulles comme le montre la figure 5.10

```
KPI1          0
KPI2          0
KPI3          0
KPI4          0
KPI5          0
KPI6          0
KPI7          0
KPI8          0
Slice Type    0
dtype: int64
```

Figure 5.10: Résultats de vérification des valeurs manquantes.

e) Encodage de données catégoriale

Comme nous avons cité dans le chapitre 3 (section 4.3.1.3) que les données de notre ensemble de données sont déjà numérique, donc nous avons utilisé que la variable fictives depuis la bibliothèque OneHotEncoder afin d'aborder le problème de corrélation

La figure 5. 11 montre les données avant et après cette étape.

```
Avant utilisée Dummy variable
[2 2 1 ... 0 0 2]

Aprés utilisée Dummy variable
[[0. 0. 1.]
[0. 0. 1.]
[0. 1. 0.]
...
[1. 0. 0.]
[1. 0. 0.]
[0. 0. 1.]]
```

Figure 5.11: Encodage de données .

f) Division de l'ensemble de données

Après avoir chargé l'ensemble de données nous allons le diviser en un échantillon d'entraînement (training set) de70% qui est utilisé par les modèles pour s'entrainer dessus et un échantillon d'évaluation (testing set) de 30% qui est inconnu pour le modèle et qui consiste à tester et à évaluer la capacité des modèles pour la prédiction de ces nouvelles données. Pour faire cela on utilise la fonction **train_test_split()** comme module Python(voir figure 5.12)

```
Nombre de transactions X_train dataset:  (145063, 8)
Nombre de transactions y_train dataset:  (145063,)
Nombre de transactions X_test dataset:  (62170, 8)
Nombre de transactions y_test dataset:  (62170,)
```

Figure 5.12: Division de l'ensemble de données en train et test

g) Mise à l'échelle des caractéristiques

Cette étape sert à normaliser les variables de l'ensemble de données dans une plage spécifique, nous avons utilisé la standardisation (voir figure 5.13)

```
Avant Standardisation :
 [[ 6 12  2 ...  2  3  1]
 [13  6  2 ...  1  2  2]
 [ 2 10  2 ...  2  1  2]
 ...
 [11 12  1 ...  1  1  5]
 [14  7  1 ...  2  3  5]
 [ 9 22  2 ...  2  3  1]]

Aprés Standardisation :
 [[-0.3698972   0.14757684  0.74300158 ...  0.6709513   1.12272074
  -0.90346372]
 [ 1.36322277 -0.75738675  0.74300158 ... -1.49042114 -0.01365407
  -0.37824604]
 [-1.36025146 -0.15407769  0.74300158 ...  0.6709513  -1.15002887
  -0.37824604]
 ...
 [ 0.86804564  0.14757684 -1.34589216 ... -1.49042114 -1.15002887
   1.19740699]
 [ 1.61081134 -0.60655948 -1.34589216 ...  0.6709513   1.12272074
   1.19740699]
 [ 0.3728685   1.6558495   0.74300158 ...  0.6709513   1.12272074
  -0.90346372]]
```

Figure 5.13: Les données avant et après la standardisation.

h) Préparation et prétraitement de données

Parmi les techniques citées dans le chapitre précédent pour traiter la distribution de classe déséquilibrée nous avons utilisé la technique de SMOTE (Synthetic Minority Oversampling Technique – Oversampling). Et elle nous donne une meilleure performance en terme de

rappel et de précision.Les figures suivantes (Figure 5.14 et Figure 5.15) montrent la distribution des classes avant et après le sur-échantillonnage.

(a)

```
Nombre de transactions X_train dataset:  (145063, 8)
Nombre de transactions y_train dataset:  (145063,)
Nombre de transactions X_test dataset:  (62170, 8)
Nombre de transactions y_test dataset:  (62170,)
```

(b)

```
Classe = 2, n=74219 (35.814%)
Classe = 1, n=51787 (24.990%)
Classe = 0, n=81227 (39.196%)
```

(c)

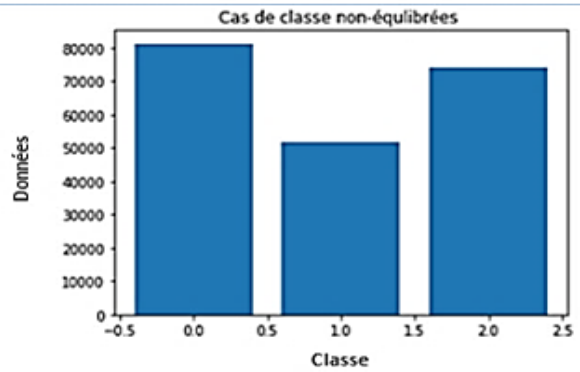


Figure 5.14: histogramme et résultats obtenues de distribution des classes avant le sur-échantillonnage.

Apres avoir appliqué l'algorithme de SMOTE a sur-échantillonne (Over-Sampling) des cas minoritaires, il les rendre égaux à la classe majoritaires. Les trois catégories ont un nombre égal d'enregistrements. Plus précisément, les catégories de la minorité ont été portées au nombre total de la catégorie de la majorité.

(a)

```
Aprés le suréchantillonnage , La forme de train_X: (170862, 8)
Aprés le suréchantillonnage , La forme de train_y: (170862,)

Aprés le suréchantillonnage, Le nombre d'étiquettes '0': 56954
Aprés le suréchantillonnage, Le nombre d'étiquettes '1': 56954
Aprés le suréchantillonnage, Le nombre d'étiquettes '2': 56954
```

(b)

```
Class=0, n=56954 (27.483%)
Class=1, n=56954 (27.483%)
Class=2, n=56954 (27.483%)
```

(c)

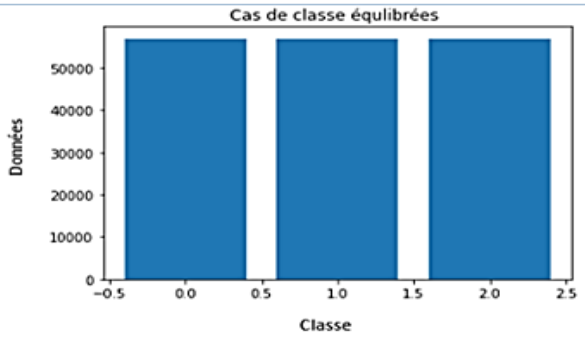


Figure 5.15: histogramme et résultats obtenues de distribution des classes aprés le sur-échantillonnage.

i) Choix des hyperparamètres

Représente le problème de choisir un ensemble d'hyperparamètres optimaux pour un algorithme d'apprentissage. L'objectif est de trouver une combinaison optimale d'hyperparamètres pour améliorer les performances du modèle.

Il existe plusieurs techniques pour le choix des hyperparamètres, pour notre travail nous avons effectué une recherche manuelle en fonction du premier choix des hyperparamètres, nous changeons par la suite l'un des hyperparamètres puis entraînons à nouveau le modèle, et vérifions les résultats obtenus en terme des métriques d'évaluation. (voir ANNEXE B)

5.5.Entrainement et évaluation des modèles

5.5.1. Algorithmes d'apprentissage automatique

a) K-NN

- **Les hyperparamètres utilisés**

Le tableau 5.3 montre les hyperparamètres utilisés pour l'entraînement de modèle KNN (voir ANNEXE A section 2.1.1)

Hyperparamètres	Valeur
n_neighbors	20
Leaf_size	7
p	10

Tableau 5.3: Les hyperparamétres utilisés pour l'entrainement de modéle KNN.

- **Résultats obtenues et discussion**

La figure 5.16 suivantes montre les résultats d'évaluation de performance d'entrainement de modèle KNN (voir ANNEXE A section 3)

```
------------------------------------ Rapport de classification --------
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     28690
           1       0.97      0.85      0.91     18300
           2       0.92      0.99      0.95     14180

    accuracy                           0.95     62170
   macro avg       0.95      0.94      0.95     62170
weighted avg       0.95      0.95      0.95     62170

------------------------------------------------------------------------

-------------------------------- Taux d'exactitude ---------------------
Taux exactitude de test classificateur KNN  : 94.94611548978608
Taux exactitude de train classificateur KNN : 96.38565844713604
------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.951072
Recall: 0.949461
F1 score: 0.948558
-->>> Selon cohens kappa
Cohens kappa: 0.923432
-->>> Selon Auc
AUC: 0.997
------------------------------------ Evaluaion de prediction ----------

MAE: 0.052
------------------------------------ Temps d'éxecution de modéle  -----

 le temps execution de modele de KNN est :0.058 seconds
------------------------------------------------------------------------
```

Figure 5.16: Résultats d'entrainement de KNN.

La matrice de confusion présente une métrique importante pour faire évaluer les modèles la figure 5.17représente la matrice obtenu par le KNN.

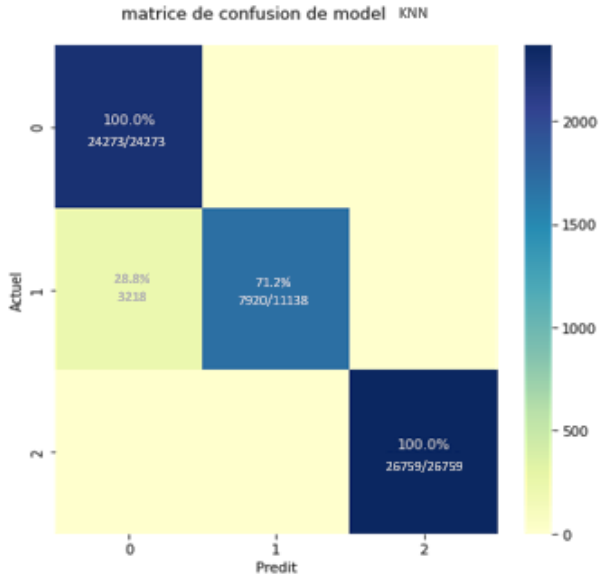


Figure 5.17: Matrice de confusion de KNN.

Discussion :

- Classe 0 (emBB) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%)
- Classe 1 (mMTC) :72.2 % des instances sont prédit correctement (taux d'exactitude 72.2% et taux d'erreur 28.8 %)
- Classe 2 (uRLLC) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%)

b) ANN

Les hyperparamètres utilisés

Le tableau 5.4 montre les hyperparamètres utilisés pour l'entraînement de modèle ANN (voir ANNEXE A section 2.1.2)

hyperparamètres	Valeurs
Nbr de neurone	3
Fonction d'activation	Relu
Solver	Adam
max-iter	300

Tableau 5.4: Les hyperparamétres utilisés pour l'entrainement de modéle ANN.

Résultats obtenus et discussion

La figure 5.18 suivantes montre les résultats l'évaluation de performance d'entrainement de modèle ANN .

```
---------------------------------- Rapport de classification  ----------
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     24273
           1       0.71      1.00      0.83     11138
           2       1.00      0.83      0.91     26759

    accuracy                           0.93     62170
   macro avg       0.90      0.94      0.91     62170
weighted avg       0.95      0.93      0.93     62170

-------------------------------------------------------------------------

-------------------------------- Accuracy --------------------------------
Taux d'exactitude de test classificateur ANN : 92.70870194627634
Taux d'exactitude  de train  de classificateur ANN   : 92.94306611610128
-------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.948178
Recall: 0.927087
F1 score: 0.929878
-->>> Selon cohens kappa
Cohens kappa: 0.887572
-->>> Selon Auc
AUC: 0.956
------------------------------------ Evaluaion de prediction ----------

MAE: 0.073
------------------------------------ Temps d'éxecution de modéle  -------

 le temps execution de modele de ANN est :10.493 seconds
```

Figure 5.18: Résultats d'entrainement de ANN.

- **Matrice de confusion**

  La matrice de confusion de modèle ANN est représentée dans la figure 5.19 suivante.

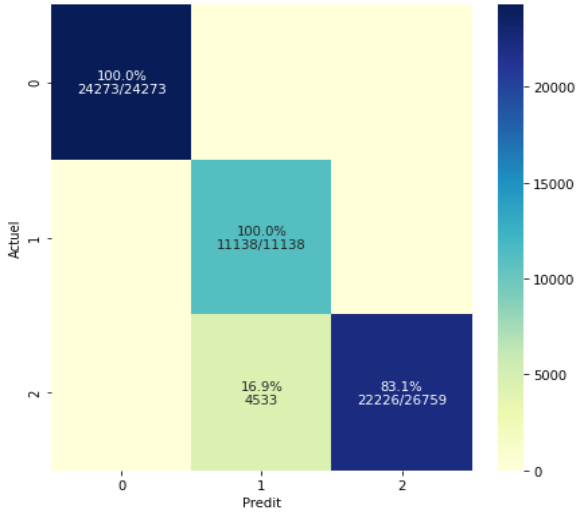


***Figure* 5.19:** Matrice de confusion de ANN.

Discussion :

- Classe 0 (emBB) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%).
- Classe 1 (mMTC) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%).
- Classe 2 (uRLLC) : 83.1% sont prédit correctement (taux d'exactitude 83.1% et taux d'erreur 16.9 %).

c) Decision Tree

Hperparamètres choisis

Le tableau 5.5 montre les hyperparamètres utilisés pour l'entraînement de modèle Decision Tree (voir ANNEXE A section 2.1.3).

hyperparamètres	Valeurs
max-iter	4

Tableau 5.5: Les hyperparamètres utilisés pour l'entrainement de modèle Decision Tree.

Résultats obtenus et discussion

La figure 5.20 suivantes montre les résultats l'évaluation de performance d'entrainement de modèle Decision Tree.

```
-------------------------Evaluation de performance de DecisionTreeClassifier -------

------------------------------------ Rapport de classification  --------------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.78      0.88     20130
           2       0.90      1.00      0.95     20053

    accuracy                           0.93     62170
   macro avg       0.94      0.93      0.92     62170
weighted avg       0.94      0.93      0.93     62170

-------------------------------------------------------------------------------------

------------------------------- Taux d'exactitude ---------------------------------
 Taux d'exactitude de test de classificateur decision tree: 0.9282773041659964
 Taux d'exactitude de train de classificateur decision tree: 0.9364106705996652
-------------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.935158
Recall: 0.928277
F1 score: 0.925618
-->>> Selon cohens kappa

Cohens kappa: 0.892145
-->>> Selon Auc

AUC: 0.994
------------------------------------ Evaluaion de prediction ----------

MAE: 0.072
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de Decision Tree est :0.114 seconds
```

Figure 5.20: Résultats d'entrainement de Decision Tree.

- **Matrice de confusion**

La matrice de confusion est représentée dans la figure 5.21 ci-dessous.

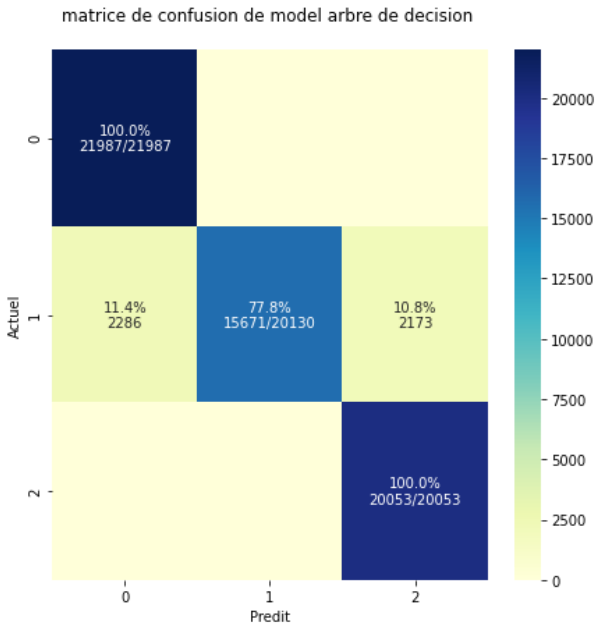


Figure 5.21: Matrice de confusion de Decision Tree.

Discussion :

- Classe 0 (emBB) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%).
- Classe 1 (mMTC) :77.8% des instances sont prédit correctement (taux d'exactitude 77.8% et taux d'erreur 22.2 %)
- Classe 2 (uRLLC) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%).

d) Random Forest

- **Résultats obtenus et discussion**

La figure 5.22 suivantes montre les résultats d'évaluation de performance d'entrainement de modèle Random Forest (voir ANNEXE A section 2.1.4)

```
-------------------------Evaluation de performance de Random Forest--------------------

------------------------------------ Rapport de classification  ----------------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.87      0.93     17957
           2       1.00      1.00      1.00     22226

    accuracy                           0.96     62170
   macro avg       0.97      0.96      0.96     62170
weighted avg       0.97      0.96      0.96     62170

-------------------------------------------------------------------------------------------

-------------------------------- Taux d'exactitude ----------------------------------------
Taux d'exactitude de test de classificateur Random Forest : 0.9632298536271514
Taux d'exactitude de train de classificateur Random Forest : 0.9693905022766911
-------------------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.966693
Recall: 0.963230
F1 score: 0.962889
-->>> Selon cohens kappa
Cohens kappa: 0.944398
-->>> Selon Auc
AUC: 1.000
------------------------------------ Evaluaion de prediction ----------

MAE: 0.037
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de Random Forest est :2.881 seconds
-----------------------------------------------------------------------
```

Figure 5.22: Résultats d'entrainement de Random Forest.

- **Matrice de confusion**

  Présenter dans la figure 5.23.

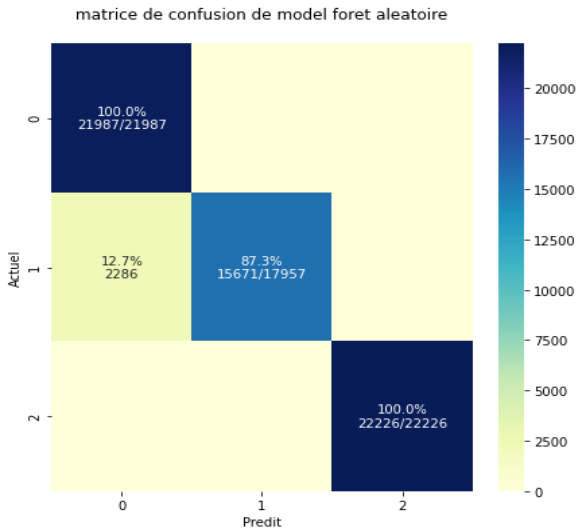


Figure 5.23: Matrice de confusion de Random Forest.

Discussion :

- Classe 0 (emBB) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%).

- Classe 1 (mMTC) :87.3% des instances sont prédit correctement (taux d'exactitude 87.3% et taux d'erreur 12.7 %)
- Classe 2 (uRLLC) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%).

e) LDA

La figure ci-dessous montre les résultats obtenus par le modèle LDA .

```
-------------------------Evaluation de performance de LinearDiscriminan

------------------------------------ Rapport de classification  -------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.87      0.93     17957
           2       1.00      1.00      1.00     22226

    accuracy                           0.96     62170
   macro avg       0.97      0.96      0.96     62170
weighted avg       0.97      0.96      0.96     62170

------------------------------------------------------------------------

------------------------------- Taux d'exactitude --------------------
Taux d'exactitude de test classificateur LDA : 0.9632298536271514
Taux d'exactitude de train de  classifier LDA : 0.9693905022766911
------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.966693
Recall: 0.963230
F1 score: 0.962889
------------------------------------------------------------------------

-->>> Selon Cohens kappa

Cohens kappa: 0.944398
------------------------------------------------------------------------

-->>> Selon AUC

AUC: 0.984
-->>> Selon temps d'execution

le temps d'execution de modele est :0.200 seconds
```

Figure 2.24: Résultats d'entrainement de LDA.

- **Matrice de confusion de LDA**

La figure 5.25 ci-dessous représente la matrice de confusion.

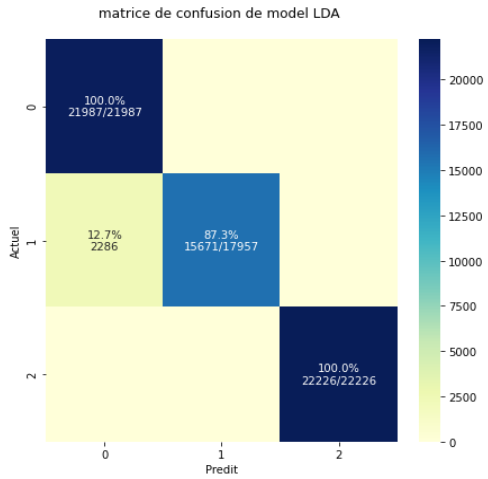


Figure 5.25: Matrice de confusion de LDA.

Discussion :

- Classe 0 (emBB) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%)
- Classe 1 (mMTC) :83.7% des instances sont prédit correctement (taux d'exactitude 83.9% et taux d'erreur 12.7 %)
- Classe 2 (uRLLC) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%)

f) Naïve bayes

La figure 5.26 ci-dessous montre les résultats obtenus par le modèle Naïve Bayes .

```
---------------------------------- Accuracy -----------------------------------------
                taux d'exactitude de test de classificateur Naive Bayes : 0.8929869712079781
                taux d'exactitude de train de classificateur Naive Bayes 0.8937082508978857
-------------------------------------------------------------------------------------

------------------------Evaluation de performance de Naive bayes --------------------------

------------------------------------ Rapport de classification  --------------------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       0.86      0.75      0.80     17936
           2       0.90      0.90      0.90     22247

    accuracy                           0.89     62170
   macro avg       0.89      0.88      0.88     62170
weighted avg       0.89      0.89      0.89     62170

-------------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.891316
Recall: 0.892987
F1 score: 0.890273
-->>> Selon cohens kappa

Cohens kappa: 0.838171
-->>> Selon AUC

AUC: 0.976
-->>> Temps d'execution

 le temps d execution de modele est :0.049 seconds
-----------------------------------------------------------------------

------------------------------------ Evaluaion de prediction ------------------------------

MAE: 0.107
-----------------------------------------------------------------------
```

Figure 5.26: Résultats d'entrainement de Naïve Bayes.

- **Matrice de confusion de modèle Naïve Bayes (voir figure 5.27)**

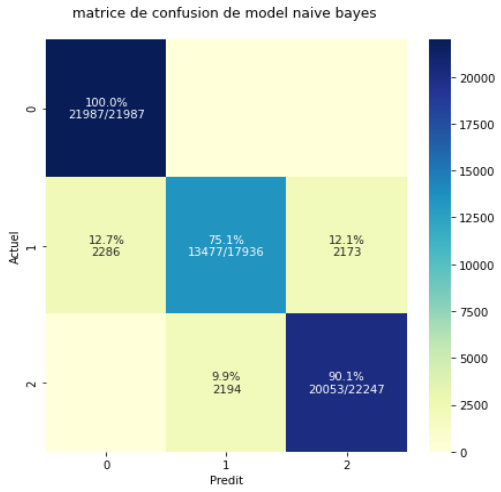


Figure 5.27: Matrice de confusion de Naïve Bayes.

Discussion :

- Classe 0 (emBB) : tous les instances sont correctement prédit (taux d'exactitude 100% et taux d'erreur 0%)
- Classe 1 (mMTC) :75.1% des instances sont prédit correctement (taux d'exactitude 83.9% et taux d'erreur 24.8 %)

- Classe 2 (uRLLC) : 90.1% sont prédit correctement (taux d'exactitude 83.1% et taux d'erreur 9.9 %)

Comparaison entre les algorithmes d'apprentissage automatique

- **La validation croisée (voir ANNEXE A section 2.2)**

```
KNN: 0.929466 (0.002074)
RF: 0.963732 (0.001911)
DT: 0.929466 (0.002074)
LDA: 0.963732 (0.001911)
NB: 0.893492 (0.003529)
ANN: 0.929466 (0.002074)
```

Figure 5.28: Résultats obtenus par la validation croisée .

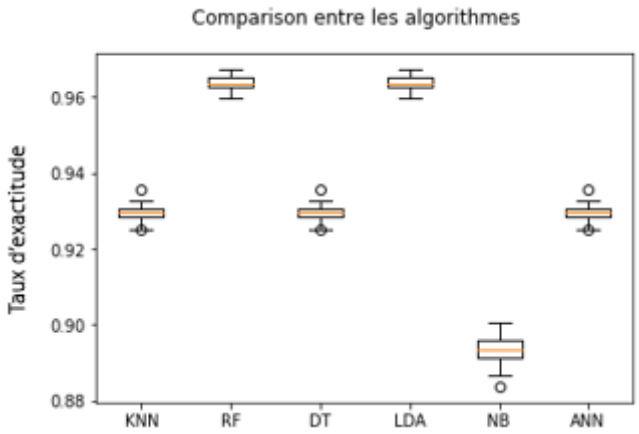


Figure 5.29: Comparaison entre les différents algorithmes.

- Depuis les matrices de confusion des diffèrent algorithmes d'apprentissage automatique, on peut comparer entre les performances des modèles selon le nombre des instances qui sont correctement classées et celle qui sont incorrectement classé comme le montre le tableau 5.7.

Nbr total d'instances	Algorithmes de classifications	correctement classées	incorrectement classé
62170	Random Forest	59884	2286
	KNN	58952	3218
	DT	57711	4459
	LDA	59884	2286
	NB	55517	6653
	ANN	57637	4533

Tableau 5.6: Performance du classificateur à l'aide de nombre des instances classifiées.

- ✓ Pour les algorithmes d'apprentissage automatique, les résultats d'entrainement ont un bon score d'utilité pour les trois classes eMBB, mMTC et uRLLC.
- ✓ Dans les tableaux 5.7, nous avons montré les résultats selon diffèrent mesure d'évaluation de la classification (précision, rappel, mesure-f, AUC, Cohen kappa, taux d'exactitude, MAE).
- ✓ Les algorithmes d'apprentissage automatique kNN, DT, LDA, ANN, NB et Random Forest ont presque tous de bonne performance trouvée par la simulation avec l'ensemble de données, et le meilleur algorithme est le RF avec un taux d'exactitude de 96.3%.

5.5.2. Apprentissage profond

a) DNN

Les hyperparamètres utilisés

Les hyperparamètres utilisés pour l'entraînement de modèle DNN (voir ANNEXE A section 2.3.1)

- **Nombre de couche cachée :** 4
- **Nombre d'époque (epochs) :**16 époques
- **Taille de lot (Batch size):**128
- **Fonction d'optimisation (Optimizer) :**Adam
- **Fonction de loss (loss) :** categorical_crossentropy
- **kernel_initializer :** random_normal

Notre modèle DNN prédit avec précision à quelle tranche de réseau on attribue un dispositif connecté au réseau. Dans la figure 5.30 nous montrons une aperçu sur le modèle DNN construit qui a quatre couches cachées est une fonction d'activation tanh.

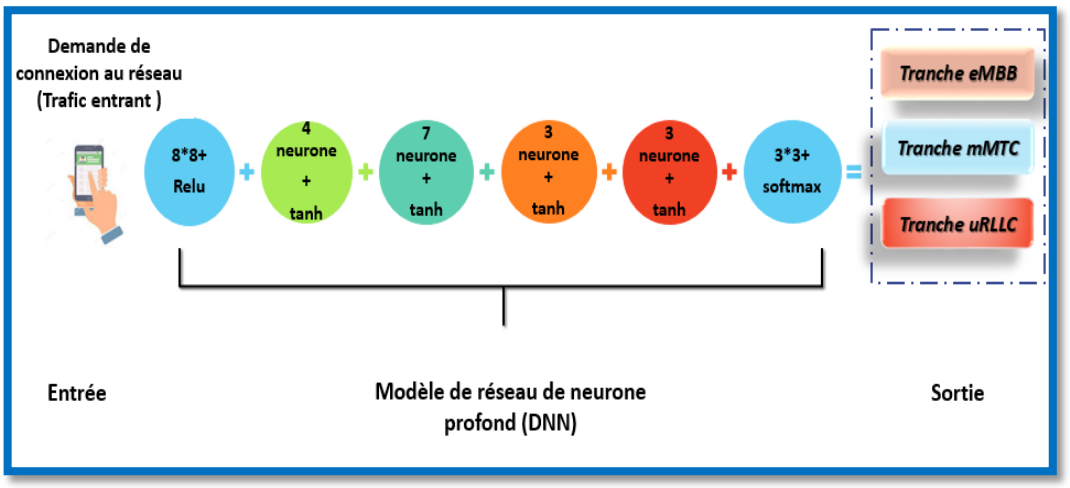


Figure 5.30: Aperçu sur le modèle DNN.

Résultats obtenus

La figure 5.31 suivantes montre les résultats d'évaluation de performance d'entrainement de modèle Deep neural network -DNN (voir ANNEXE A section 2.3)

```
--------------->>> Evaluation de performance de DNN >>>---------------

-->>> Rapport de classification

              precision    recall  f1-score   support

           0       0.91      1.00      0.95     24273
           1       1.00      0.85      0.92     15671
           2       1.00      1.00      1.00     22226

    accuracy                           0.96     62170
   macro avg       0.97      0.95      0.96     62170
weighted avg       0.97      0.96      0.96     62170

-->>> Matrice de confusion

[[24273     0     0]
 [ 2339 13332     0]
 [    0     0 22226]]


 Taux d'exactitude de DNN est : 96.51%
-->>> Selon MAE

MAE: 0.046
-->>> Selon Auc

AUC: 0.969
-->>>  temps d'execution

 le temps d'execution de modele est :41.847 seconds
```

Figure 5.31: Résultats d'entrainement de modèle DNN.

Selon la matrice de confusion on trouve :

- Classe 0 (emBB) : toutes les instances sont correctement prédites.

- Classe 1 (mMTC) :13332 parmi 15671 sont correctement prédit.
- Classe 2 (uRLLC) : toutes les instances sont correctement prédites.

Nbr total	Correctement classifier	Incorrectement classifier
62170	59831	2339

Tableau 5.8 : Performance du classificateur à l'aide de nombre des instances classifiées

La figure 5.32 illustre la convergence de notre modèle DNN.

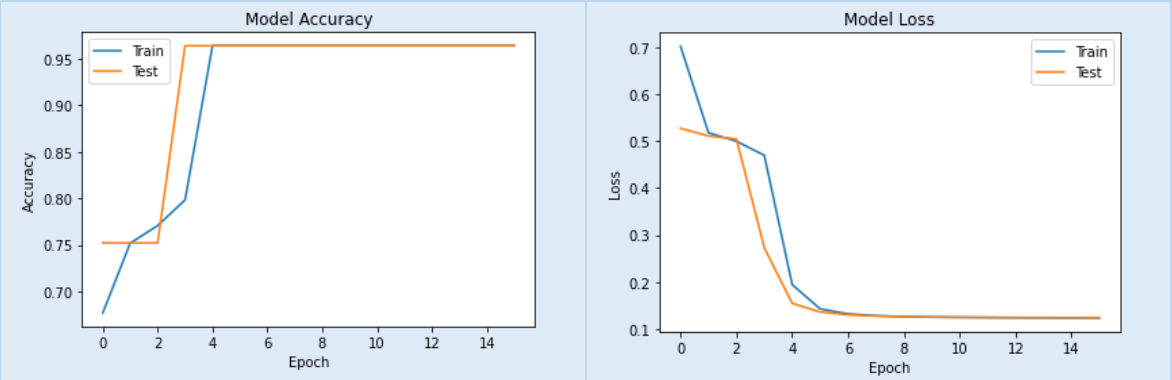


Figure 5.32: entrainement et test taux d'exactitude et perte.

Après avoir testé plusieurs algorithmes d'apprentissage automatique et le réseau de neurone profond, on remarque que la plupart des algorithmes donnent des bons résultats, mais le plus optimal et qui répond le mieux à notre problématique c'est le DNN qui nous a donné un meilleur taux d'exactitude de 96.51% comparé aux autres algorithmes(voire tableau 5.7) testés : Random Forest, ANN, DT, LDA, KNN et NB qui ont donné un taux d'exactitude de 96.3%, 92.7%,92.8% et 96.3%,94.9% et 89.8% respectivement.

	NB	LDA	RF	KNN	DT	ANN	Meilleure modèle ML	DNN	Meilleure modèle
Taux d'exactitude	89.2%	96.3%	96.3%	94.9%	92.8%	92.7%	96.3%	96.51%	96.51%
Précision	89%	97%	96%	94%	93%	94%	96%	97%	97%
Rappel	89%	96%	96%	94%	92%	92%	96%	96%	96%
F-mesure	89%	96%	96%	94%	92%	92%	96%	96%	96%
AUC	0.976	0.984	1	0.997	0.994	0.956	1	0.969	0.969
Cohens Kappa	0.838	0.944	0.944	0.923	0.892	0.887	0.944	/	/
MAE	0.107	0.984	0.037	0.107	0.072	0.073	0.037	0.046	0.046
Temps d'execution	0.049	0.200	2.881	0.049	0.114	10.493	2.881	41.84	41.84
							RF		**DNN**

Tableau 5.7 : Comparaison entre les modèles

5.6. Comparaison avec les modèles des travaux connexes

Pour comparer notre modèle DNN avec les travaux connexes cité en chapitre 2 on a utilisé le taux d'exactitude et taux de perte qui sont exprimé aussi dans les figure 5.30 et 5.31 comme métriques de comparaison, le tableau résume les résultats de notre modèle et les deux autres travaux [36], [37] où on remarque que notre modèle a donné de meilleures performances par rapport à ces derniers.

Travail	Description de l'architecture utilisée	Taux d'exactitude	Taux de perte
[38]	**4+relu** + **4+Relu** + **3+tanh** + **3+softmax**	68,89%	31.11%
[1]	**5+relu** + **8+Relu** + **4+tanh** + **3+softmax**	95%	5%
Notre modèle	Voir figure 5.30	96.51%	3.49%

Tableau 5.8: Comparaison avec les travaux connexes.

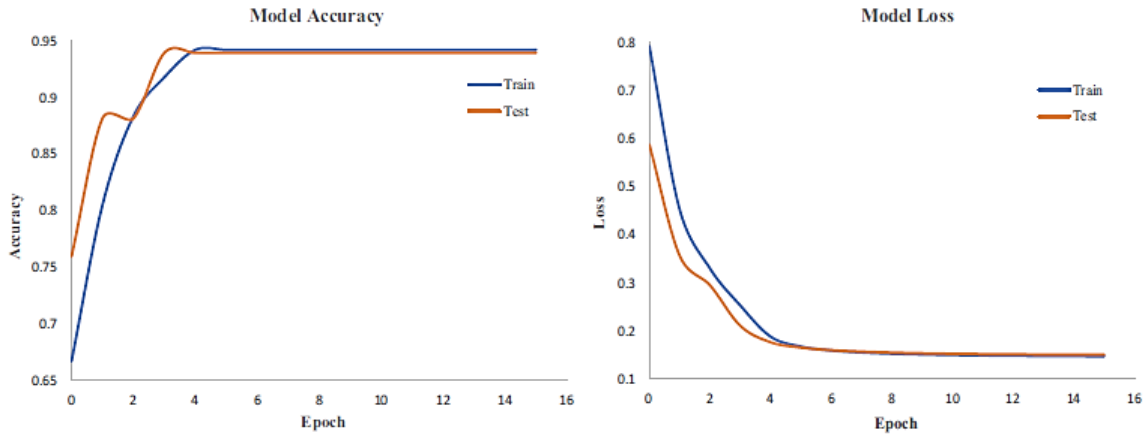


Figure 5.33: Taux d'exactitude et de perte d'entrainement et de test. [2]

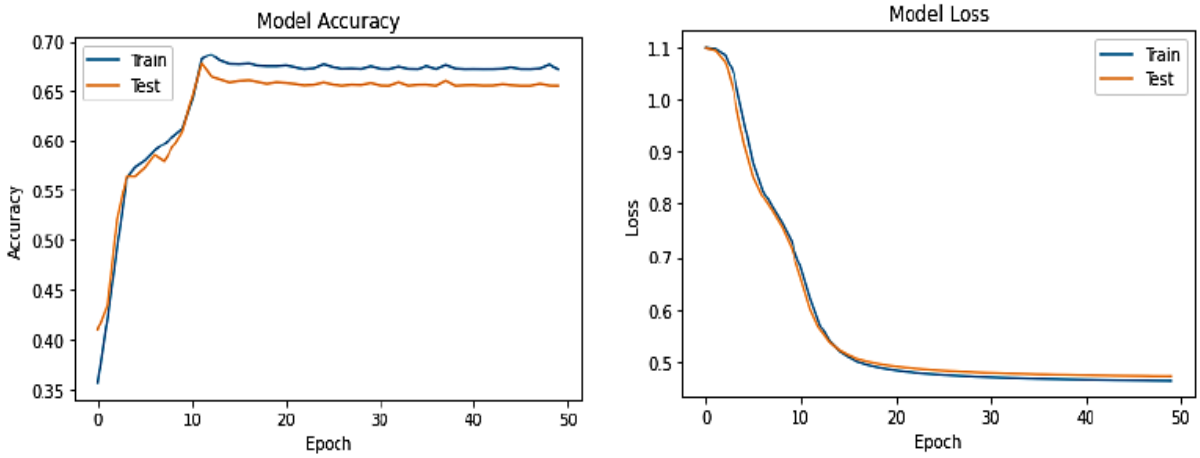


Figure 5.34: Taux d'exactitude et de perte d'entrainement et de test. [38]

5.7. Conclusion

Dans ce chapitre nous avons commencé par la description de l'ensemble de données et exprimer l'ensemble de données après le prétraitement et les outils et l'environnement utilisée pour réaliser ce travail.

Nous avons par la suite présenté les résultats de notre implémentation des algorithmes d'apprentissage automatique LDA,RF, NB, ANN , DT et le KNN et d'apprentissage profond en utilisant le DNN et en finir par l'étude comparative avec les travaux connexes ou notre modèles a de meilleur performance par rapport à ces derniers .

Conclusion Générale

La 5G est la technologie de rupture, qui est doté de la capacité de satisfaite les exigences de ces trois catégories d'usage ; le haut débit mobile évoluée (eMBB), la communication de type machine massive (mMTC) et la communication ultra-fiable et à faible latence (uRLLC).

Le découpage de réseaux (Network Slicing) est considéré comme une technologie clé où chaque tranche de réseau est adaptée à chaque catégorie d'usages selon les indicateurs de performances (KPIs) requit.

Dans ce travail, nous traitons la sélection efficace des tranches réseau pour n'importe quel type d'équipement connecté dépendamment des indicateurs de performance clés (KPI) en utilisons les techniques d'apprentissage automatique et le réseau de neurone profond

Nous avons en premier lieu analysé l'ensemble de données et son prétraitement, en deuxième lieu on a implémenté de nombreux algorithmes d'apprentissage automatique dont : Random Forest, LDA, KNN, DT, NB, ANN. Tous ces algorithmes on des bonnes performances en terme des différent métriques d'évaluation utilisée en particulier avec le Random forest et LDA. Par la suite, nous avons construit un modèle basé sur les réseaux de neurones profond qui a son tour donné de meilleure performance que les modèles d'apprentissage automatique implémentés.

Limitation

Malgré la qualité des résultats obtenus mais le choix manuelle des hyperparamètres des modèles proposés à travers, plusieurs tests en variant manuellement les différents hyperparamètres des modèles présente une limite importante dans générations du modèle adéquat .

Perceptives

Comme perspectives de recherches futures :

- Nous envisageons ajouter des métadonnées tel que des métriques de routages afin d'augmenter l'ensemble de données utilisé pour un découpage de réseau plus fiable et efficace.
- Implémenter des technologies pour le choix automatique des hyperparamètres.

Bibliographie

[1] Kafle, Ved ,P. Fukushima, Yusuke, Martinez-Julia, Pedro, Et al . *«Consideration on automation of 5G network slicing with machine learning »*. In : 2018 ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K) . IEEE, 2018. p. 1-8.

[2] Thantharate, Anurag, Paropkari, Rahul, Walunj, Vijay, et al . « Deepslice: A deep learning approach towards an efficient and reliable network slicing in 5G networks » . In : 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) . IEEE, 2019. p. 0762-0767.

[3] Fabrizio Granelli, Chapter 3 - Network slicing,Editor(s): Frank H.P. Fitzek, Fabrizio Granelli, Patrick Seeling . *«Computing in Communication Networks»* . Academic Press, 2020,Pages.

[4] Arcep, *«Les enjeux de la 5G»*. Site Web : https://www.arcep.fr/la-regulation/grands-dossiers-reseaux-mobiles/la-5g.html . Mars 2017.Publier le 24 août 2018 . Consulté le 05/04/2021.

[5] Frederic Launay. «Le réseau 5G – 5GS» . Site Web : http://blogs.univ-poitiers.fr/flaunay/2018/08/24/le-reseau-5g-5gs/ .Consulté le 15/02/2021.

[6] 5G white paper . «5G Vision , Requirements , and Enabling Technologies». Deliverable, vol. 2.0, pp. 10–14, 2016.

[7] Assane Ngom , «Conception de petits réseaux d'antennes reconfigurables ou ' SmallCells ' pour le standard 5G ». 2019. these doctorat, l'Université Nice Côte d'Azur.

[8] Popovski, Petar, Trillingsgaard, Kasper Floe, Simeone, Osvaldo, et al . *« 5G wireless network slicing for eMBB, URLLC, and mMTC: A communication-theoretic view »* . IEEE Access, 2018, vol. 6, p. 55765-55779.

[9] Series, M . *«IMT Vision–Framework and overall objectives of the future development of IMT for 2020 and beyond »* . Recommendation ITU, 2015, vol. 2083.

[10] Stephen ,M. Blust . *« En route vers les IMT-2020 (5G) »* . Recommendation ITU, 2017.

[11] S. Talwar. D. Choudhury, K. Dimou, E. Aryafar, B. Bangerter And K. Stewart. *«Enabling technologies and architectures for 5G wireless »* . 2014 IEEE MTT-S International Microwave Symposium (IMS2014), Tampa, FL, USA, 2014, pp. 1-4 .

[12] S. Zhang. *«An Overview of Network Slicing for 5G »* . in IEEE Wireless Communications, 2017.

[13] Chataut, Robin Et Akl, Robert . *« Massive MIMO systems for 5G and beyond networks—overview, recent trends, challenges, and future research direction»* . Sensors, 2020, vol. 20, no 10, p. 2753.

[14] Kim, Dongwook Et Kim, Sungbum . *«Network slicing as enablers for 5G services: state of the art and challenges for mobile industry»*. Telecommunication Systems, 2019, vol. 71, no 3, p. 517-527.

[15] S. Zhang . *«* An Overview of Network Slicing for 5G*»* . in IEEE Wireless Communications, vol. 26, no. 3, pp. 111-117, June 2019.

[16] Zhu, Guosheng, Zan, Jun, Yang, Yang, et al . *«A supervised learning based QoS assurance architecture for 5G networks»* . IEEE Access, 2019, vol. 7, p. 43598-43606.

[17] Morocho-Cayamcela, Manuel Eugenio, Lee, Haeyoung, Et Lim, Wansu . *«Machine learning for 5G/B5G mobile and wireless communications: Potential, limitations, and future directions»*. IEEE Access, 2019, vol. 7, p. 137184-137206.

[18] Liu, Yuxi Hayden . *« Python Machine Learning By Example »* . Packt Publishing Ltd, 2017.

[19] Fourati, Hasna, Maaloul, Rihab, Et Chaari, Lamia, *« A survey of 5G network systems: challenges and machine learning approaches »*. International Journal of Machine Learning and Cybernetics, 2021, vol. 12, no 2, p. 385-431.

[23] Osisanwo, F. Y., Akinsola, J. E. T., Awodele, O., et al . *« Supervised machine learning algorithms: classification and comparison »* . International Journal of Computer Trends and Technology (IJCTT), 2017, vol. 48, no 3, p. 128-138.

[24] Kavish Sanghvi , *« Image Classification Techniques »* ,
URL :https://medium.com/analytics-vidhya/image-classification-techniques-83fd87011cac Publié le 8 Mai , 2020,Consulté le 15/05/2020.

[25] Choudhary, Rishabh Et Gianey, Hemant Kumar . *«Comprehensive review on supervised machine learning algorithms»*. In : 2017 International Conference on Machine Learning and Data Science (MLDS). IEEE, 2017. p. 37-43.

[26] A. Lundervold And Ar. Lundervold . *«An overview of deep learning in medical imaging focusing on mri»* . Zeitschrift fur Medizinische Physik, 29, 12 2018.

[27] Bouafia Nadjat. *«Classification efficace des vêtements de mode basée sur les approches : apprentissage automatique ML et apprentissage profond DL»* .2017.Mémoire master.Universite Mohamed Boudiaf - M'sila.

[28] Afroz Chakure , *« K-Nearest Neighbors Algorithm »*,

Site Web :https://medium.datadriveninvestor.com/k-nearest-neighbors-knn-algorithm bd375d14eec7

Publié le 09 janvier 2019, Consulté le 05/04/2021.

[29] Laoubi Mohamed Chems Eddine. *« Contrôle d'un pendule inversé par un réseau de neurones artificiels »*. 2018.Mémoire de Master. Université Akli Mohand Oulhadj – Bouira.

[30] Pirmin Lemberger, *«Deep Learning pas à pas»*,

Site Web : https://www.technologies-ebusiness.com/enjeux-et-tendances/le-deep-learning-pas-a-pas

Publié le 28 mai 2015, Consulté le 15/05/2021.

[31] Das, Himanish Shekhar Et Roy, Pinki . *« A deep dive into deep learning techniques for solving spoken language identification problems In: Intelligent Speech Signal Processing»*. Academic Press, 2019. p. 81-100.

[32] Debbabi, Fadoua, Jmal, Rihab, Fourati, Lamia Chaari, et al. *«Algorithmics and Modeling Aspects of Network Slicing in 5G and Beyonds Network: Survey»*. IEEE Access, 2020, vol. 8, p. 162748-162762.

[33] Fabrizio Granelli.Chapter 3 - Network slicing,Editor(s): Frank H.P. Fitzek, Fabrizio Granelli, Patrick Seeling . *« Computing in Communication Networks »*. Academic Press, 2020,Pages.

[34] Deng, Li Et Yu, Dong . *« Deep learning: methods and applications, Foundations and trends in signal processing »*. 2014, vol. 7, no 3–4, p. 197-387.

[35] Pranoy Radhkrishnan , *«what are Hyperparameters ? and how to tune the Hyperparameters in a Deep Neural Network ? »* ,

Site Web : https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917584a

Publié le 09 aout 2017 . Consulté le 29/04/2021.

[36] Kafle, Ved P., Martinez-Julia, Pedro, Et Miyazawa, Takaya . *« Automation of 5G network slice control functions with machine learning »* . IEEE Communications Standards Magazine ,2019, vol. 3, no 3, p. 54-62.

[37] Mcclellan, Miranda, Cervelló-Pastor, Cristina, Et Sallent, Sebastià . *« Deep Learning at the Mobile Edge: Opportunities for 5G Networks »*. Applied Sciences, 2020, vol. 10, no 14, p. 4735.

[38] Gupta, Rohit Kumar Et Misra, Rajiv. *«Machine Learning-based Slice allocation Algorithms in 5G Networks»*. In : 2019 International Conference on Advances in Computing, Communication and Control (ICAC3). IEEE, 2019. p. 1-4.

[39] Gupta, Rohit Kumar, Pannu, Praduman, Et Misra, Rajiv. *«Towards ultra-latency using deep learning in 5G Network Slicing applying Approximate k-Nearest Neighbor Graph Construction»*.2021.

[40] *« Dataset-Unicauca-Version2 »* ,
Site web: https://www.kaggle.com/kerneler/starter-ip-network-traffic-flows-49778383-1
Consulté le 10/05/2021.

[41] *« DeepSlice dataset »*,
SiteWeb : https://www.kaggle.com/anuragthantharate/deepslice.
Consulté 20/31/2021.

[42] Site Web : https://www.datacamp.com/community/tutorials/data-preparation-with-pandas .Consulté le 03/04/2021.

[43] Site Web: https://hackernoon.com/what-steps-should-one-take-while-doing-datapreprocessing-502c993e1caa.
Consulté le 03/04/2021.

[44] Abraham, Shiny, Huynh, Chau, Et Vu, Huy. *«Classification of soils into hydrologic groups using machine learning»*. Data, 2020, vol. 5, no 1, p. 2.

[45] Laqrichi, Safae .*«Approche pour la construction de modèles d'estimation réaliste de l'effort/coût de projet dans un environnement incertain: application au domaine du développement logiciel »*.2015.Thèse de doctorat . Ecole nationale des Mines d'Albi-Carmaux.

ANNEXE A : Code Source

1. Prétraitement de données

1.1.Mise à l'échelle de données

```
# Standardisation
Standardisation = preprocessing.StandardScaler()
x_after_Standardisation = Standardisation.fit_transform(X)
print ("\nAvant Standardisation : \n", X)
print ("\nAprés Standardisation : \n", x_after_Standardisati \n)
```

1.2.Equilibrage de données

```
#visualisation de distribution de classe avant SMOTE
counter = Counter(y)
for k,v in counter.items():
per = v / len(y) * 100
print('Classe = %d, n=%d (%.3f%%)' % (k, v, per))
pyplot.bar(counter.keys(), counter.values())
pyplot.title("Cas de classe non-équlibrées ")
pyplot.show()
# Décrire les informations de train and test set
print("Nombre de transactions X_train dataset: ", X_train.shape)
print("Nombre de transactions y_train dataset: ", y_train.shape)
print("Nombre de transactions X_test dataset: ", X_test.shape)
print("Nombre de transactions y_test dataset: ", y_test.shape)
from imblearn.over_sampling import SMOTE
oversample = SMOTE()
X_train_res, y_train_res = oversample .fit_sample(X_train, y_train.ravel())
 #Visualisation de distribution de classe après SMOTE
counter = Counter(y_train_res)
for k,v in counter.items():
per = v / len(y) * 100
print('Class=%d, n=%d (%.3f%%)' % (k, v, per))
pyplot.bar(counter.keys(), counter.values())
pyplot.show()
```

1.3.Encodage de données catégorielles

```
from sklearn.preprocessing import  OneHotEncoder
encoder = LabelEncoder()
encoder.fit(y)
encoded_Y = encoder.transform(y)
dummy_y = np_utils.to_categorical(encoded_Y)
```

2. Entrainement des modèles

2.1.Les modèles d'apprentissage automatique

2.1.1. KNN

```
KNN_model = KNeighborsClassifier(n_neighbors = 20,leaf_size=1,p=3)
```

2.1.2. ANN

```
from sklearn.neural_network import MLPClassifier
MLP_model =  MLPClassifier(hidden_layer_sizes=(8,), max_iter=300,activation = 'relu',s
olver='adam')
start = time()
MLP_model.fit(X_train_res, y_train_res)
end = time()
```

2.1.3. Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
DT_model = DecisionTreeClassifier(max_depth = 4)
start = time()
DT_model.fit(X_train_res, y_train_res)
end = time()
```

2.1.4. Random Forest

```
from sklearn.ensemble import RandomForestClassifier
RF_model  =RandomForestClassifier(n_estimators=150,max_depth=4,random_state=5,ma
x_samples=100,min_samples_split=3)
start = time()
RF_model.fit(X_train_res, y_train_res)
end = time()
```

2.1.5. LDA

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
LDA_model=LinearDiscriminantAnalysis()
start = time()
LDA_model.fit(X_train_res, y_train_res)
end = time()
```

2.1.6. Naïve Bayes

```
from sklearn.naive_bayes import GaussianNB
NB_model = GaussianNB()
start = time()
NB_model.fit(X_train_res, y_train_res)
end = time()
```

2.2. Validation des modèles

```
models = []
models.append(('KNN', KNeighborsClassifier(n_neighbors = 20,leaf_size=1,p=3)))
models.append(('RF', RandomForestClassifier(n_estimators=150,max_depth=4,random_st
ate=5,max_samples=100,min_samples_split=3)))
models.append(('DT', DecisionTreeClassifier(max_depth = 4)))
models.append(('LDA',LinearDiscriminantAnalysis()))
models.append(('NB', GaussianNB()))
models.append(('ANN',MLPClassifier(hidden_layer_sizes=(8,), max_iter=300,activation =
 'relu',solver='adam')))
# Evaluation de chaque modèle
results = []
names = []
scoring = 'accuracy'

from sklearn.metrics import recall_score
for name, model in models:
  kfold = model_selection.KFold(n_splits=30, random_state=7,shuffle=True)
  cv_results = model_selection.cross_val_score(model, X,Y, cv=kfold, scoring=scoring)

  results.append(cv_results)

  names.append(name)
  msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
 print(msg)
```

2.3.Les modèles d'apprentissage profond

2.3.1. DNN

```
classifier = Sequential()
#La couche d'entrer
classifier.add(Dense(8, activation='relu', kernel_initializer='random_normal', input_dim=8)
)
#1er couche cachée
classifier.add(Dense(4, activation='tanh', kernel_initializer='random_normal'))
#2eme couche cachée
classifier.add(Dense(7, activation='tanh', kernel_initializer='random_normal'))
#3eme couche cachée
classifier.add(Dense(3, activation='tanh', kernel_initializer='random_normal'))
```

```
#4eme couche cachée
classifier.add(Dense(3, activation='tanh', kernel_initializer='random_normal'))
#Couche de sortie
classifier.add(Dense(3, activation='softmax', kernel_initializer='random_normal'))
```

3. Script de visualisation d'évaluation de performance (exemple de script de visualisation de Decision Tree)

3.1.1. La matrice de confusion

```
def plot_cm(DT_prediction, y_test, figsize=(7,7)):
    cm = confusion_matrix(DT_prediction, y_test, labels=np.unique(y_test))
    cm_sum = np.sum(cm, axis=1, keepdims=True)
    cm_perc = cm / cm_sum.astype(float) * 100
    annot = np.empty_like(cm).astype(str)
    nrows, ncols = cm.shape
    for i in range(nrows):
        for j in range(ncols):
            c = cm[i, j]
            p = cm_perc[i, j]
            if i == j:
                s = cm_sum[i]
                annot[i, j] = '%.1f%%\n%d/%d' % (p, c, s)
            elif c == 0:
                annot[i, j] = ''
            else:
                annot[i, j] = '%.1f%%\n%d' % (p, c)
    cm = pd.DataFrame(cm, index=np.unique(y_test), columns=np.unique(y_test))
    cm.index.name = 'Actuel'
    cm.columns.name = 'Predit'
    fig, ax = plt.subplots(figsize=figsize)
    sns.heatmap(cm, cmap= "YlGnBu", annot=annot, fmt='', ax=ax)

plot_cm(DT_prediction, y_test)
plt.title('\n matrice de confusion de model arbre de decision  \n')
```

- **Exemple d'éxecution**

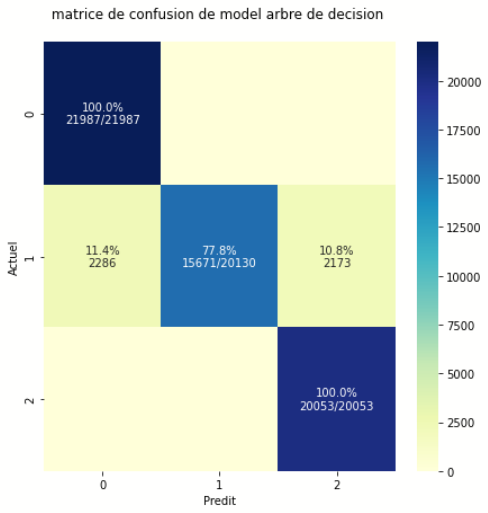


Figure 1: Matrice de confusion de modèle DT

- **Tracage de convergence de modéle**

```
from sklearn.model_selection import validation_curve
k=np.arange(1,5)
train_score , val_score =validation_curve(DT_model,X_test,y_test,'max_depth',k,cv=30)
plt.plot(k, val_score.mean(axis=1),label='test')
#plt.plot(k, train_score.mean(axis=1),label='train')
plt.xlabel('score')
plt.xlabel('max_depth')
plt.legend()
```

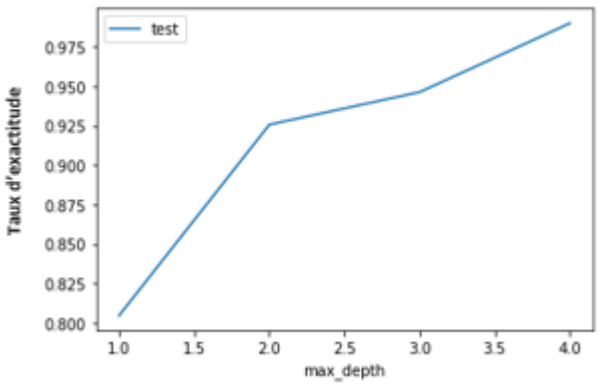


Figure 2: Convergence de modèle DT

3.1.2. Les métriques d'évaluation de performance (exemple de script d'évaluation de performances de KNN)

```
print("-->>> Selon la précision , rappel et f-measure \n ")
# precision tp / (tp + fp)
precision = precision_score(DT_prediction, y_test,average='weighted')
print('Precision: %f' % precision)
# recall: tp / (tp + fn)
recall = recall_score(DT_prediction, y_test,average='weighted')
print('Recall: %f' % recall)
# f1: 2 tp / (2 tp + fp + fn)
f1 = f1_score(DT_prediction, y_test,average='weighted')
print('F1 score: %f' % f1)
print("-->>> Selon cohens kappa  \n ")
# kappa
kappa = cohen_kappa_score(DT_prediction, y_test)
print('Cohens kappa: %f' % kappa)
print("-->>> Selon Auc \n ")
y_pred= DT_model.predict_proba(X_test)
# calculate AUC
auc = roc_auc_score(y_test, y_pred,multi_class='ovr')
print('AUC: %.3f' % auc)
print("------------------------------------ Evaluaion de prediction ----------\n")
# evaluate predictions
mae = mean_absolute_error(y_test, DT_prediction)
print('MAE: %.3f' % mae)
print("------------------------------------ Temps d'éxecution de modéle  ----------\n")
# report execution time
result = end - start
print(' le temps execution de modele de Decision Tree est :%.3f seconds' % result)
```

- **Exemple d'exécution**

```
----------Evaluation de performance de DecisionTreeClassifier -------

------------------------------------ Rapport de classification  --------------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.78      0.88     20130
           2       0.98      1.00      0.99     20053

    accuracy                           0.93     62170
   macro avg       0.94      0.93      0.92     62170
weighted avg       0.94      0.93      0.93     62170

------------------------------------------------------------------------------------

------------------------------- Taux d'exactitude -------------------------------
 Taux d'exactitude de test de classificateur decision tree: 0.9282773041659964
 Taux d'exactitude de train de classificateur decision tree: 0.9364106705996652
------------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.935158
Recall: 0.928277
F1 score: 0.925618
-->>> Selon cohens kappa

Cohens kappa: 0.892145
-->>> Selon Auc

AUC: 0.994
------------------------------------ Evaluaion de prediction ----------

MAE: 0.072
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de Decision Tree est :0.114 seconds
```

Figure 3 : Evaluation de performance de DT

ANNEXE B : Résultats supplémentaires des différents tests

1. Résultats supplémentaires des différents tests

1.1. KNN

Le choix de la valeur à utiliser pour effectuer une prédiction avec K-NN, varie en fonction de l'ensemble de données. En règle générale, moins on utilisera de voisins (un nombre petit) plus on sera sujette au sous apprentissage (underfitting). Par ailleurs, plus on utilise de voisins (un nombre K grand) plus, sera fiable dans notre prédiction.

En a Essayer plusieurs combinaisons en variant le nombre k et faire du tuning de l'algorithme pour avoir un résultat satisfaisant. Le tableau ci-dessous résume les résultats de quelques tests effectués pour arriver à les hyperparametres optimisé.

Les hyperparamètres utiliser	Expérimentation 1	Expérimentation 2	Expérimentation 3
n_neighbors	25	30	20
Leaf_size	1	7	7
p	3	10	10

- **Résultats de 1er test :**

```
------------------------------------ Rapport de classification -------
              precision    recall  f1-score   support

           0       0.91      0.98      0.94     22141
           1       0.97      0.78      0.87     13270
           2       0.87      0.95      0.91     26759

    accuracy                           0.91     62170
   macro avg       0.91      0.90      0.90     62170
weighted avg       0.91      0.91      0.91     62170

----------------------------------------------------------------------

------------------------------- Taux d'exactitude --------------------
Taux exactitude de test classificateur KNN  : 90.87984558468715
Taux exactitude de train classificateur KNN : 92.3418423973363
----------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.913192
Recall: 0.908798
F1 score: 0.907101
-->>> Selon cohens kappa
Cohens kappa: 0.862425
-->>> Selon Auc
AUC: 0.991
------------------------------------ Evaluaion de prediction ---------

MAE: 0.107
------------------------------------ Temps d'éxecution de modéle  ----

 le temps execution de modele de KNN est :0.042 seconds
----------------------------------------------------------------------
```

- **Résultats de 2éme test**

```
---------------------------------- Rapport de classification  -------
              precision    recall  f1-score   support

           0       0.93      0.99      0.96     22141
           1       0.97      0.82      0.89     10270
           2       0.91      0.96      0.94     28759

    accuracy                           0.93     62170
   macro avg       0.93      0.92      0.93     62170
weighted avg       0.93      0.93      0.93     62170

------------------------------------------------------------------------

------------------------------- Taux d'exactitude --------------------
Taux exactitude de test classificateur KNN  : 93.0030561364002
Taux exactitude de train classificateur KNN : 94.64845823481156
------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.932387
Recall: 0.930031
F1 score: 0.928626
-->>> Selon cohens kappa
Cohens kappa: 0.894241
-->>> Selon Auc
AUC: 0.995
---------------------------------- Evaluaion de prediction ---------

MAE: 0.078
---------------------------------- Temps d'éxecution de modéle  ----

 le temps execution de modele de KNN est :0.058 seconds
------------------------------------------------------------------------
```

- **Résultats de 3éme test**

```
---------------------------------- Rapport de classification  --------
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     28690
           1       0.97      0.85      0.91     18300
           2       0.92      0.99      0.95     14180

    accuracy                           0.95     62170
   macro avg       0.95      0.94      0.95     62170
weighted avg       0.95      0.95      0.95     62170

------------------------------------------------------------------------

------------------------------- Taux d'exactitude ---------------------
Taux exactitude de test classificateur KNN  : 94.94611548978608
Taux exactitude de train classificateur KNN : 96.38565844713604
------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.951072
Recall: 0.949461
F1 score: 0.948558
-->>> Selon cohens kappa
Cohens kappa: 0.923432
-->>> Selon Auc
AUC: 0.997
---------------------------------- Evaluaion de prediction ----------

MAE: 0.052
---------------------------------- Temps d'éxecution de modéle  -----

 le temps execution de modele de KNN est :0.058 seconds
------------------------------------------------------------------------
```

➢ **Discussion :**

- Notre modèle de KNN final prend les hyperparametres de 1er test qui ne donne de bonnes performances par rapport aux autres tests effectués comme le montre le tableau ci-dessous.

	Taux d'exactitude de Test	Précision	Rappel	F-mesure	Choens Kappa	MAE	AUC	Temps d'exécution
Test 1	90%	91%	90%	90%	0.86	0.107	0.991	0.042
Test 2	93%	93%	93%	92%	0.89	0.078	0.995	0.058
Test3	94%	95%	94%	92.8%	0.92	0.052	0.997	0.058

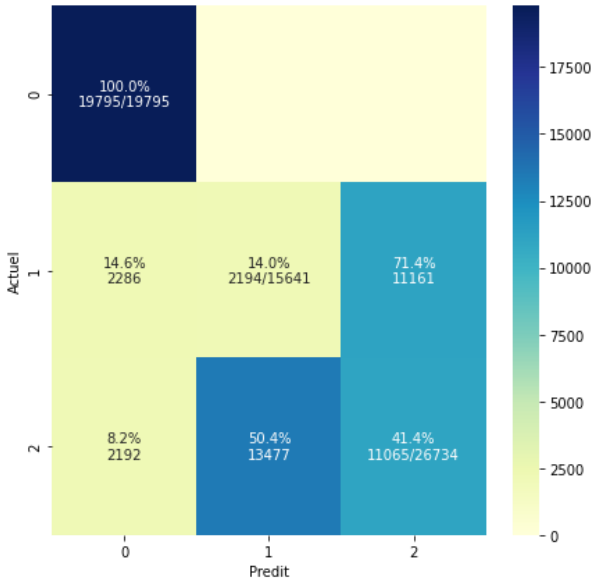
1.2. ANN

Pour le ANN en a fixé le nombre des couches cachées a une seule couche et en a varier le nombre de neurone de cette couche en obtient les résultats suivant :

Les hyperparamètres utiliser	Test 1	Test 2	Test 3	Test 4
Nbr de neurone	8	2	3	3
Fonction d'activation	relu	relu	relu	tanh
Solver	Adam	Adam	Adam	Adam
max-iter	300	300	300	300

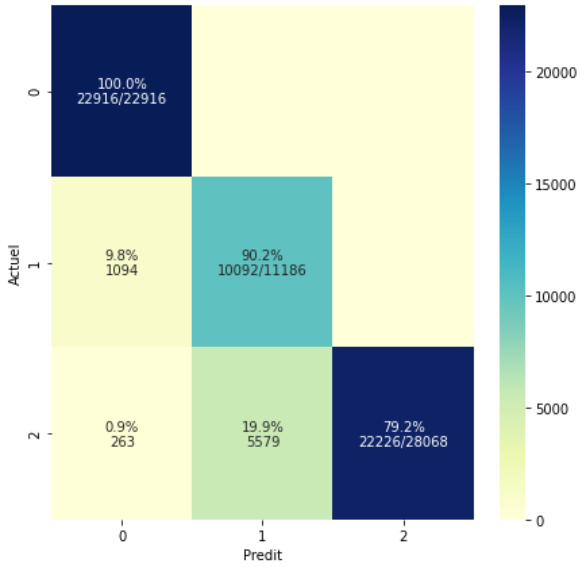
- **Résultats de 1er test**

```
------------------------------------ Rapport de classification  -----------
              precision    recall  f1-score   support

           0       0.82      1.00      0.90     19795
           1       0.14      0.14      0.14     15641
           2       0.50      0.41      0.45     26734

    accuracy                           0.53     62170
   macro avg       0.48      0.52      0.50     62170
weighted avg       0.51      0.53      0.52     62170

--------------------------------------------------------------------------------

-------------------------------- Accuracy --------------------------------
Taux d'exactitude de test classificateur ANN : 53.16712240630529
Taux d'exactitude  de train  de classificateur ANN   : 53.47331848920813
--------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.508962
Recall: 0.531671
F1 score: 0.515670
-->>> Selon cohens kappa
Cohens kappa: 0.288837
-->>> Selon Auc
AUC: 0.801
------------------------------------ Evaluaion de prediction ----------

MAE: 0.504
------------------------------------ Temps d'éxecution de modéle  --------

 le temps execution de modele de ANN est :12.606 seconds
--------------------------------------------------------------------------
```



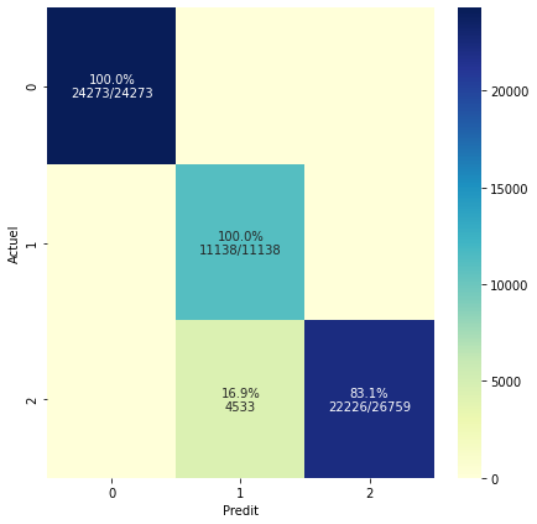
- **Résultats de 2éme test**

```
-------------------------Evaluation de performance de ANN ----------------

------------------------------------ Rapport de classification  ----------
              precision    recall  f1-score   support

           0       0.94      1.00      0.97     22916
           1       0.64      0.90      0.75     11186
           2       1.00      0.79      0.88     28068

    accuracy                           0.89     62170
   macro avg       0.86      0.90      0.87     62170
weighted avg       0.92      0.89      0.89     62170

-----------------------------------------------------------------------------

-------------------------------- Accuracy -----------------------------------
Taux d'exactitude de test classificateur ANN : 88.84349364645328
Taux d'exactitude  de train  de classificateur ANN   : 88.98409656494076
-----------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.915338
Recall: 0.888435
F1 score: 0.892253
-->>> Selon cohens kappa
Cohens kappa: 0.828185
-->>> Selon Auc
AUC: 0.994
------------------------------------ Evaluaion de prediction ----------

MAE: 0.116
------------------------------------ Temps d'éxecution de modéle  -------

 le temps execution de modele de ANN est :10.641 seconds
-----------------------------------------------------------------------
```



- **Résultat de 3éme test**

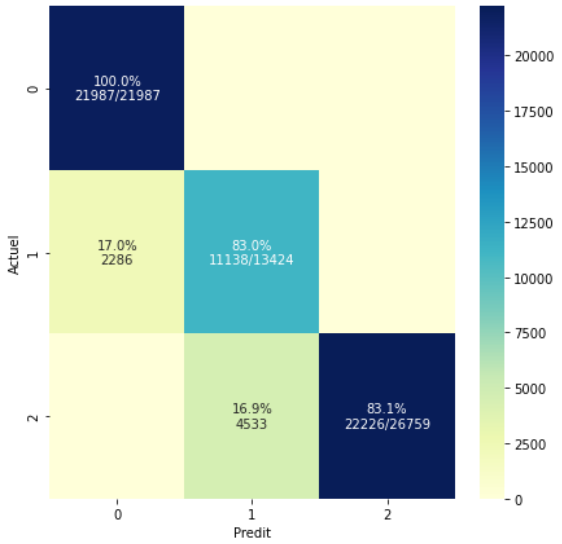
```
---------------------------------- Rapport de classification ----------
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     24273
           1       0.71      1.00      0.83     11138
           2       1.00      0.83      0.91     26759

    accuracy                           0.93     62170
   macro avg       0.90      0.94      0.91     62170
weighted avg       0.95      0.93      0.93     62170

------------------------------------------------------------------------

------------------------------- Accuracy --------------------------------
Taux d'exactitude de test classificateur ANN : 92.70870194627634
Taux d'exactitude  de train  de classificateur ANN   : 92.94306611610128
------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.948178
Recall: 0.927087
F1 score: 0.929878
-->>> Selon cohens kappa
Cohens kappa: 0.887572
-->>> Selon Auc
AUC: 0.956
------------------------------------ Evaluaion de prediction ----------

MAE: 0.073
------------------------------------ Temps d'éxecution de modéle  -------

 le temps execution de modele de ANN est :10.493 seconds
```



- **Résultats 4éme test**

Dans ce test nous avons varié la fonction d'activation où on a utilisé la fonction tanh avec 3 neurones.

```
-------------------------Evaluation de performance de ANN ----------------

------------------------------------ Rapport de classification  ----------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       0.71      0.83      0.77     13424
           2       1.00      0.83      0.91     26759

    accuracy                           0.89     62170
   macro avg       0.87      0.89      0.87     62170
weighted avg       0.90      0.89      0.89     62170

-------------------------------------------------------------------------

------------------------------- Accuracy --------------------------------
Taux d'exactitude de test classificateur ANN : 89.03168730899147
Taux d'exactitude  de train  de classificateur ANN   : 89.33773601814384
-------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.904235
Recall: 0.890317
F1 score: 0.892087
-->>> Selon cohens kappa
Cohens kappa: 0.832191
-->>> Selon Auc
AUC: 0.987
------------------------------------ Evaluaion de prediction ----------

MAE: 0.110
------------------------------------ Temps d'éxecution de modéle  -------

 le temps execution de modele de ANN est :9.804 seconds
-------------------------------------------------------------------------
```



Discussion :

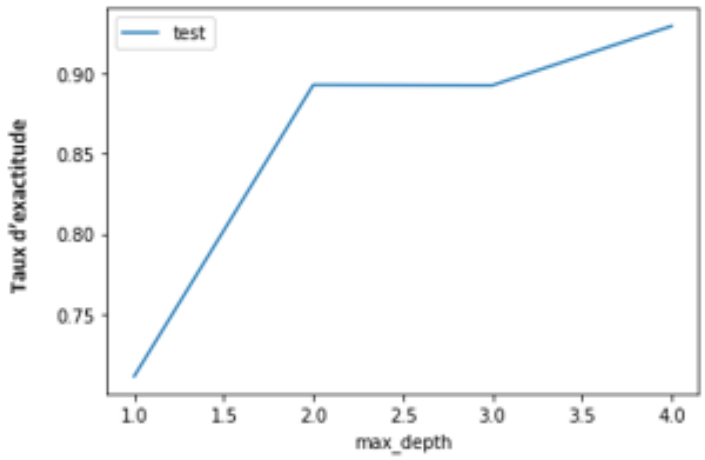
- Après avoir effectué plusieurs test notre modèle ANN prend les hyperparametres utiliser dans le 3éme test car ils ne donne de meilleur performances par rapport aux autres tests effectués comme le montre le tableau ci-dessous.

	Taux d'exactitude de Test	Précision	Rappel	F-mesure	Choens Kappa	MAE	AUC	Temps d'exécution
Test 2	53%	50%	53%	51%	0.28	0.504	0.801	12.606
Test 1	88%	91%	88%	89%	0.82	0.116	0.994	10.641
Test3	92%	94%	94%	92%	0.88	0.073	0.956	10.493
Test4	89%	90%	89%	89%	0.83	0.110	0.987	9.804

1.3. Décision Tree

L'hyperparamètre max_depth contrôle la complexité globale d'un arbre de décision. Cet hyperparamètre permet d'obtenir un compromis entre un arbre de décision sous-ajusté (underfitted) et sur-ajusté(overfitted).

Pour le choix la valeur de max_depth qui nous donne de meilleur performance en visualiser let taux d'exactitude de test en fonction de cette hyperparametre , depuis le graphe obtenu on peut conclure que les valeur max_depth=3 et max_depth=4 avec un taux d'exactitudeד environ 90 à 92%.

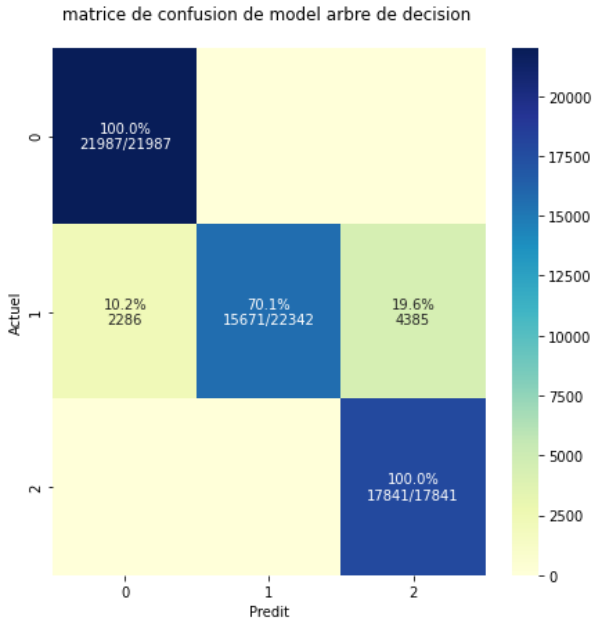


Nous allons tester notre modèle Decision Tree avec les hyperparmetre suivante

Les hyperparamètres Utiliser	Test 1	Test 2
max-iter	3	4

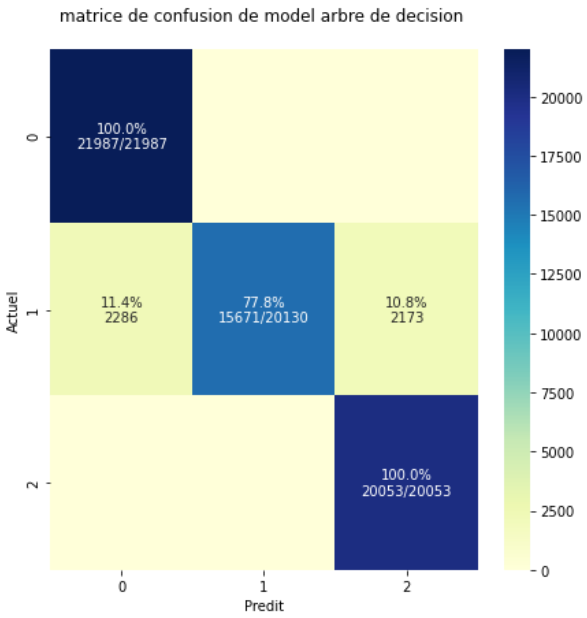
- **Résultats de 1er test**

```
-------------------------Evaluation de performance de DecisionTreeClassifier -----

------------------------------------ Rapport de classification  ------------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.70      0.82     22342
           2       0.80      1.00      0.89     17841

    accuracy                           0.89     62170
   macro avg       0.90      0.90      0.89     62170
weighted avg       0.91      0.89      0.89     62170

-------------------------------------------------------------------------------

-------------------------------- Taux d'exactitude ------------------------------
 Taux d'exactitude de test de classificateur decision tree: 0.8926974424963809
 Taux d'exactitude de train de classificateur decision tree: 0.9028046025447437
-------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.910076
Recall: 0.892697
F1 score: 0.888050
-->>> Selon cohens kappa

Cohens kappa: 0.839546
-->>> Selon Auc

AUC: 0.982
------------------------------------ Evaluaion de prediction ----------

MAE: 0.107
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de ANN est :0.100 seconds
```



- **Résultats de 2éme test**

```
-------------------------Evaluation de performance de DecisionTreeClassifier -------

------------------------------------ Rapport de classification  --------------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.78      0.88     20130
           2       0.90      1.00      0.95     20053

    accuracy                           0.93     62170
   macro avg       0.94      0.93      0.92     62170
weighted avg       0.94      0.93      0.93     62170

-------------------------------------------------------------------------------------

------------------------------- Taux d'exactitude --------------------------------
 Taux d'exactitude de test de classificateur decision tree: 0.9282773041659964
 Taux d'exactitude de train de classificateur decision tree: 0.9364106705996652
-------------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.935158
Recall: 0.928277
F1 score: 0.925618
-->>> Selon cohens kappa

Cohens kappa: 0.892145
-->>> Selon Auc

AUC: 0.994
------------------------------------ Evaluaion de prediction ----------

MAE: 0.072
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de Decision Tree est :0.114 seconds
```



➢ **Discussion :**

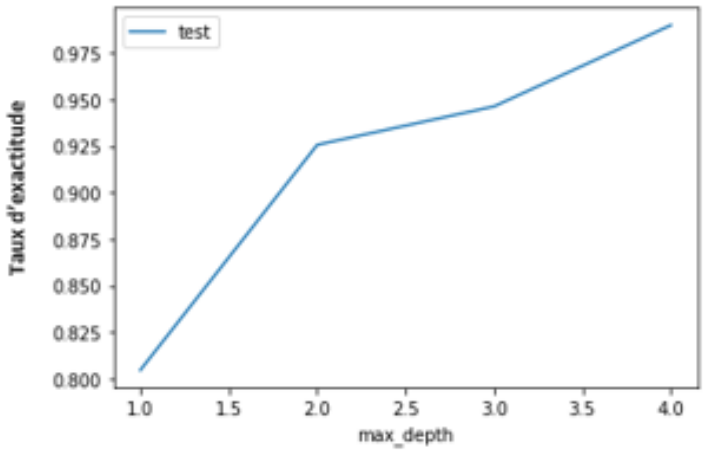
- Notre modèle de Decision Tree final prend les hyperparametres de 2éme test qui ne donne de bonnes performances par rapport aux autres tests effectués selon les différentes métriques comme le montre le tableau ci-dessous.

	Taux d'exactitude de Test	Précision	Rappel	F-mesure	Choens Kappa	MAE	AUC	Temps d'exécution
Test 1	89%	91%	89%	88%	0.83	0.107	0.982	0.1
Test 2	92%	93%	92%	92%	0.89	0.072	0.994	0.114

1.4. RF

Comme le Decision Tree le max_depth est un hypérparamétre important pour le Random forest ,un autre hyperpapamétres important pour ce algorithme est le nombre des n_estimators

Pour le choix la valeur de max_depth qui nous donne de meilleur performance en visualiser let taux d'exactitude de test en fonction de cette hyperparametre , depuis le graphe obtenu on peut conclure que les valeur max_depth=4.

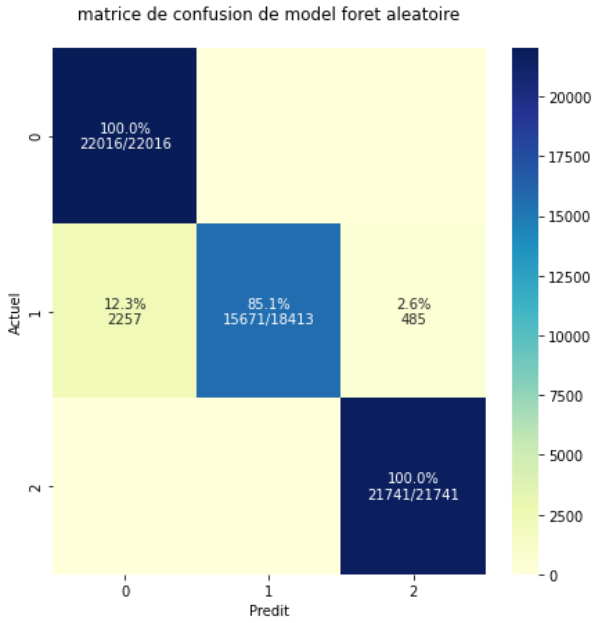


Pour le choix de la valeurs d'hperparamétres n_estimators en a fixé la valeur de p à 4 puis en varié la valeur n_estemators afin de trouver la meilleur valeur de ce dernier qui donne des performances satisfants comme le montre le tableau ci –dessous.

Les hyperparamètres utiliser	Test 1	Test 2	Test 3
`n_estimators`	50	100	150
`max_depth`	4	4	4

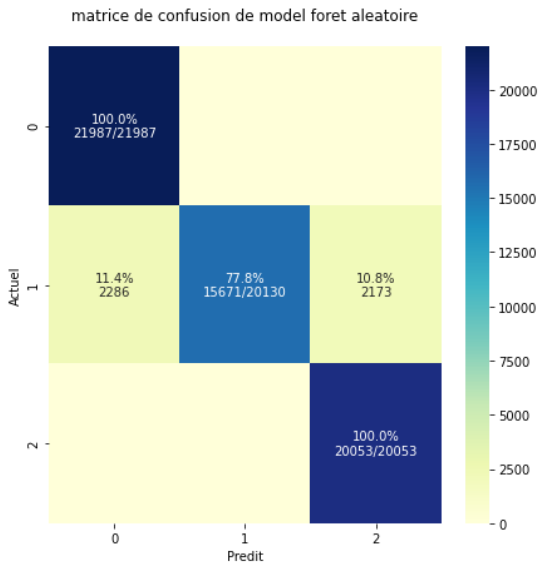
- **Résultats de 1er test**

```
-------------------------Evaluation de performance de Random Forest--------------

------------------------------------ Rapport de classification  -----------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     22016
           1       1.00      0.85      0.92     18413
           2       0.98      1.00      0.99     21741

    accuracy                           0.96     62170
   macro avg       0.96      0.95      0.95     62170
weighted avg       0.96      0.96      0.96     62170

-------------------------------------------------------------------------------

-------------------------------- Taux d'exactitude ------------------------------
Taux d'exactitude de test de classificateur Random Forest : 0.9558951262666882
Taux d'exactitude de train de classificateur Random Forest : 0.962531165501984
-------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.959441
Recall: 0.955895
F1 score: 0.955049
-->>> Selon cohens kappa
Cohens kappa: 0.933383
-->>> Selon Auc
AUC: 0.999
------------------------------------ Evaluaion de prediction ----------

MAE: 0.044
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de Random Forest est :0.979 seconds
-------------------------------------------------------------------------
```



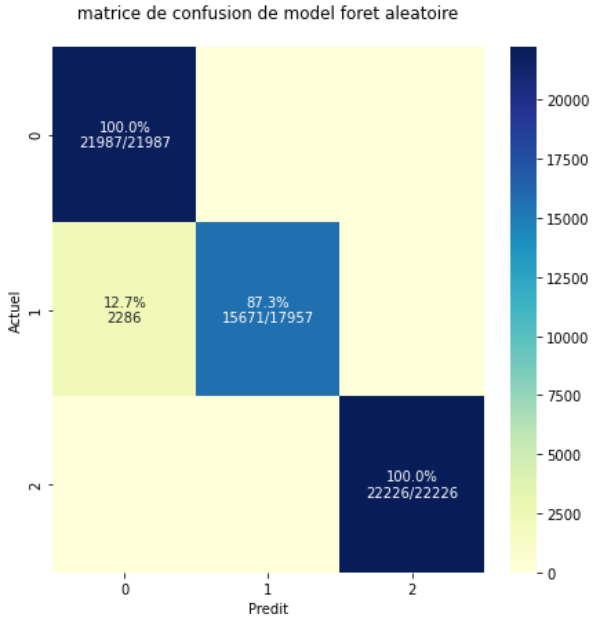
- **Résultats 2éme test**

```
---------------------------------- Rapport de classification  -----------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.78      0.88     20130
           2       0.90      1.00      0.95     20053

    accuracy                           0.93     62170
   macro avg       0.94      0.93      0.92     62170
weighted avg       0.94      0.93      0.93     62170

-------------------------------------------------------------------------------

------------------------------- Taux d'exactitude -----------------------------
Taux d'exactitude de test de classificateur Random Forest : 0.9282773041659964
Taux d'exactitude de train de classificateur Random Forest : 0.9362702063653708
-------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.935158
Recall: 0.928277
F1 score: 0.925618
-->>> Selon cohens kappa
Cohens kappa: 0.892145
-->>> Selon Auc
AUC: 0.993
------------------------------------ Evaluaion de prediction ----------

MAE: 0.072
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de Random Forest est :1.923 seconds
-------------------------------------------------------------------------
```



- **Résultats de 3éme test**

```
-------------------------Evaluation de performance de Random Forest--------------------

------------------------------------ Rapport de classification  -----------------------
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     21987
           1       1.00      0.87      0.93     17957
           2       1.00      1.00      1.00     22226

    accuracy                           0.96     62170
   macro avg       0.97      0.96      0.96     62170
weighted avg       0.97      0.96      0.96     62170

-------------------------------------------------------------------------------------------

------------------------------- Taux d'exactitude -----------------------------------
Taux d'exactitude de test de classificateur Random Forest : 0.9632298536271514
Taux d'exactitude de train de classificateur Random Forest : 0.9693905022766911
-------------------------------------------------------------------------------------------

-->>> Selon la précision , rappel et f-measure

Precision: 0.966693
Recall: 0.963230
F1 score: 0.962889
-->>> Selon cohens kappa
Cohens kappa: 0.944398
-->>> Selon Auc
AUC: 1.000
------------------------------------ Evaluaion de prediction ----------

MAE: 0.037
------------------------------------ Temps d'éxecution de modéle  ----------

 le temps execution de modele de Random Forest est :2.881 seconds
-------------------------------------------------------------------------
```



➢ **Discussion :**

	Taux d'exactitude de Test	Précision	Rappel	F-mesure	Choens Kappa	MAE	AUC	Temps d'exécution
Test 1	95%	95%	95%	95%	0.93	0.044	0.99	0.97
Test 2	92%	93%	92%	92%	0.89	0.072	0.993	1.923
Test3	96.3%	96%	96%	96%	0.94	0.037	1	2.881

1.5. Réseau de neurone artificiel (DNN-Deep neural network)

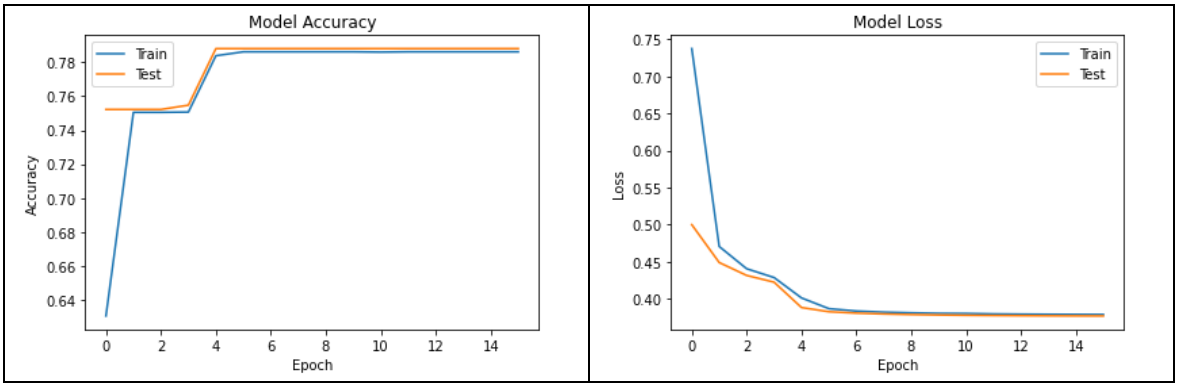
Le tableau ci-dessous montre les hyperparameters utilisée durant nos tests

Hypérparametrs	Test1	Test2	Test 3
Nombre de couche cachée	3	4	4
Nombre d'époque (epochs)	100	16	16
Taille de lot (Batch size)	256	128	128
Fonction d'optimisation (Optimizer)	Adam	Adam	Adam
Fonction de loss (loss)	categorical_crossentropy	categorical_crossentropy	categorical_crossentropy
kernel_initializer	random_normal	random_normal	random_normal

- **Résultats de 1er test**

```
--------------->>> Evaluation de performance de DNN >>>---------

-->>> Rapport de classification

              precision    recall  f1-score   support

           0       1.00      1.00      1.00     24273
           1       1.00      0.15      0.26     15671
           2       0.63      1.00      0.77     22226

    accuracy                           0.79     62170
   macro avg       0.88      0.72      0.68     62170
weighted avg       0.87      0.79      0.73     62170

-->>> Matrice de confusion

[[24273     0     0]
 [    0  2339 13332]
 [    0     0 22226]]
/usr/local/lib/python3.7/dist-packages/keras/engine/sequential.
  warnings.warn('`model.predict_proba()` is deprecated and '
-->>> Selon Taux d'exactitude


 Taux d'exactitude de DNN est accuracy: 78.56%
-->>> Selon MAE

MAE: 0.179
-->>> Selon Auc

AUC: 0.877
-->>>  temps d'execution

 le temps d'execution de modele est :41.630 seconds
```

- **Convergence de modèle**

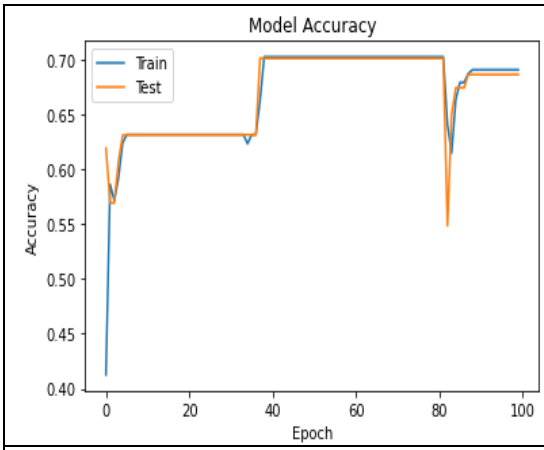


(a) (b)

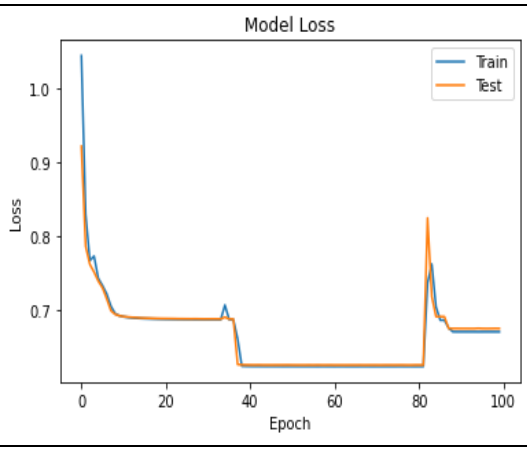
- **Résultats de 2éme test**

```
--------------->>> Evaluation de performance de DNN >>>---------------

-->>> Rapport de classification

              precision    recall  f1-score   support

           0       1.00      0.39      0.56     24273
           1       0.54      1.00      0.70     15671
           2       0.75      0.80      0.78     22226

    accuracy                           0.69     62170
   macro avg       0.76      0.73      0.68     62170
weighted avg       0.80      0.69      0.67     62170

-->>> Matrice de confusion

[[ 9468  8971  5834]
 [    0 15671     0]
 [    0  4385 17841]]
/usr/local/lib/python3.7/dist-packages/keras/engine/sequential.py:425:
  warnings.warn('`model.predict_proba()` is deprecated and '
-->>> Selon Taux d'exactitude


 Taux d'exactitude de DNN est accuracy: 69.13%
-->>> Selon MAE

MAE: 0.278
-->>> Selon Auc

AUC: 0.788
-->>>  temps d'execution

 le temps d'execution de modele est :81.981 seconds
```

- **Convergence de modèle**



(a)

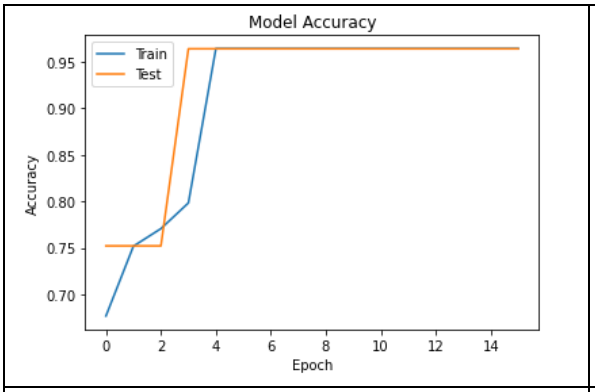


(b)

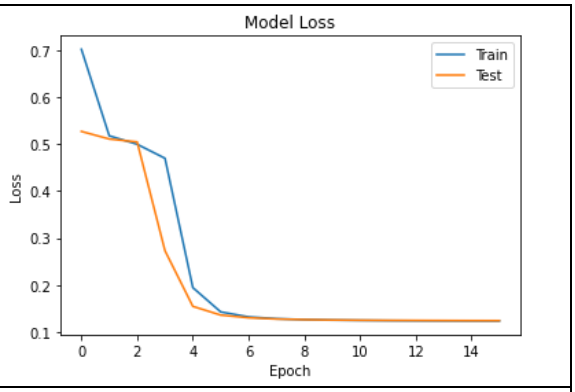
- **Résultats de 3éme test**

```
--------------->>> Evaluation de performance de DNN >>>---------------

-->>> Rapport de classification

              precision    recall  f1-score   support

           0       0.91      1.00      0.95     24273
           1       1.00      0.85      0.92     15671
           2       1.00      1.00      1.00     22226

    accuracy                           0.96     62170
   macro avg       0.97      0.95      0.96     62170
weighted avg       0.97      0.96      0.96     62170

-->>> Matrice de confusion

[[24273     0     0]
 [ 2339 13332     0]
 [    0     0 22226]]
/usr/local/lib/python3.7/dist-packages/keras/engine/sequential.py:425: UserWarning
  warnings.warn('`model.predict_proba()` is deprecated and '
-->>> Selon Taux d'exactitude


 Taux d'exactitude de DNN est : 96.51%
-->>> Selon MAE

MAE: 0.046
-->>> Selon Auc

AUC: 0.969
-->>>  temps d'execution

 le temps d'execution de modele est :41.847 seconds
```

- **Convergence de modèle**



(a)



(b)

➢ **Discussion :**

D'après les résultats obtenus en effectuant plusieurs test sur le modèles DNN , en arrivée a que les hyperparamétres utilisée en le 3éme test donne de meilleur performances en termes des métriques d'évaluation utilisée par rapport aux autre test .

	Taux d'exactitude de Test	Précision	Rappel	F-mesure	MAE	AUC	Temps d'exécution
Test 1	78.56%	87%	79%	73%	0.179	0.877	41.63
Test 2	69.13%	80%	69%	67%	0.278	0.788	81.89
Test3	96.51%	97%	96%	96%	0.046	0.969	41.847