République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou

Faculté de : Génie électrique et d'informatique
Département : Informatique

Mémoire de fin d'études

En vue de l'obtention du diplôme de
Master en Informatique

Spécialité : Systèmes Informatiques

Présenté par

ADADA L'yasmine et **HAMIDACHE Nesrine**

Thème : Déanonymisation de clients dans le réseau Bitcoin à l'aide de l'apprentissage automatique

Proposé et dirigé par : Dr.SADI Samy

Soutenue le 29/09/2020 devant le jury composé de :

Mme. BOUARAB Farida	Présidente du Jury
Mme. BERKANE Tassadit	Examinatrice
Mr. SADI Samy	Directeur de mémoire

Dédicaces

A nos chers parents,

Que nulle dédicace ne puisse exprimer ce que nous leurs devons, pour leur bienveillance, leur affection, leur patience, leur sacrifices, leur amour et leur soutien. Trésors de bonté, de générosité et de tendresse, en témoignage de notre profond amour et notre grande reconnaissance. Que dieu leurs procure bonne santé et longue vie.

A nos chères sœurs,

En témoignage de nos sincères reconnaissances pour les efforts qu'elles ont consenties pour l'accomplissement de nos études. Nous leur dédions ce modeste travail en témoignage de notre grand amour et notre gratitude infinie.

A tous nos ami(e)s,

Pour leur aide et leur support dans les moments difficiles durant l'élaboration de ce travail, avec qui on a partagé des moments inoubliables.

A nos deux chers familles ADADA et HAMIDACHE

A tous ceux dont l'oubli du nom n'est guère celui du cœur...

Remerciements

C'est avec un grand plaisir que nous réservons ces quelques lignes en signe de gratitude et de profonde reconnaissance à tous ceux qui, de près ou de loin, ont contribué à la réalisation et l'aboutissement de ce travail.

Nous tenons tout d'abord à exprimer toute notre gratitude, reconnaissance et nos remerciements à notre encadreur ***Dr. SADI Samy*** *pour son suivi, ses orientations, ses directives précieuses, et tout le temps qu'il nous a consacré tout au long de la réalisation de ce travail.*

Nous tenons aussi à exprimer l'honneur qui nous est fait par les membres du jury, en acceptant de juger notre travail.

Nous nous acquittons, enfin, d'un devoir de gratitude et de remerciements à tous nos enseignants pour la qualité de l'enseignement qu'ils ont bien voulu nous prodiguer durant nos études afin de nous fournir une formation efficiente.

Résumé

L'évolution technologique a conduit à la création et l'émergence d'une nouvelle forme de monnaie : la monnaie virtuelle ou digitale, baptisée aussi crypto-monnaie.
Ce travail concerne la dé-anonymisation des transactions *bitcoin*, l'une des crypto-monnaies les plus utilisées actuellement, en exploitant les algorithmes d'apprentissage automatique, plus précisément, les réseaux de neurones artificiels. Etant donné que le *bitcoin* permet des transactions rapides et sécurisées, néanmoins, son niveau d'anonymat devient de plus en plus faible avec l'évolution potentielle des autres différentes technologies. Notre contribution consiste à formuler un modèle d'apprentissage automatique puissant et fiable pour établir la classe de chaque trasactin Bitcoin, en utilisant des données étiquetés qui décrivent ces transitions.
Avant ce faire, nous avons effectués, en premier lieu, une étude globale concernant l'apprentissage automatique et la *blockchain*, en jetant un œil sur les différents types, caractéristiques et algorithmes utilisés pour mieux comprendre le fonctionnement de chaque technologie. Pour dé-anonymiser le client *Bitcoin*, nous proposons trois modèles de réseaux de neurones profond supervisé, qui prennent en entrée les caractéristiques des transactions collectées par la société *Elliptic* dans la plateforme *kaggle*. Ces transactions sont classées selon deux classes *(licite/illicite)* par les modèles réalisés. Entre autres, nous avons implémenté et évalué ces modèles pour effectuer, par la suite, une comparaison des résultats obtenus afin de tirer le mieux adapté en termes de temps et de puissance.

Mots clés : apprentissage automatique, réseau de neurones, blockchain, crypto-monnaie virtuelle, Bitcoin, dé-anonymisation, Tensorflow, Keras.

Abstract

Technological development has led to the creation and emergence of a new form of currency : virtual or digital currency, also known as cryptocurrency. This work concerns the deanonymization of bitcoin transactions, one of the most widely used cryptocurrencies today, by exploiting machine learning algorithms, more specifically, artificial neural networks. While bitcoin allows fast and secure transactions, its level of anonymity gets lower with the potential evolution of other different technologies. In our contribution, we formulate a powerful and reliable machine learning model to associate each transaction with its real identity, using labeled data that describes these transactions.
Before doing so, we first presented general concepts and state-of-the-art advancements on machine learning and blockchain, taking a look at the different types, characteristics and algorithms used to better understand how each technology works. To de-anonymize a bitcoin client, we propose three supervised deep neural network models, which take as input the transactions' features collected by the Elliptic company in the kaggle platform. These transactions are classified into two classes (licite/illicite) by the models we produced. Among other things, we have implemented and evaluated these models to subsequently compare the obtained results to optimize our models.

Keywords : machine learning, neural network, blockchain, virtual cryptocurrency, Bitcoin, de-anonymization, Tensorflow, Keras.

.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

1 Contexte du travail

Intelligence artificielle (*IA*), données massives (*Big data*) et *blockchain*, des termes ou mots clés, à la une, qui ont révolutionné le monde technologique ces dernières années. La digitalisation exponentielle et massive de toutes les activités humaines, dans tous les domaines, notamment le domaine économique, avec l'apparition de la crypto monnaie (la monnaie virtuelle), pousse les chercheurs à développer des nouvelles idées et expériences, pour garantir un environnement plus sécurisé pour la mobilité des données et le partage sélectif d'informations, sans révéler l'identité de la personne ou le passage par le tiers de confiance, en assurant ainsi l'intégrité des données. Ceci est dû à l'utilisation des différentes technologies complexes, dont la cryptographie, qui a connu une forte médiatisation dans la résolution des problèmes de cybersécurité, notamment par les plateformes et les fonds de crypto-monnaies.

Aujourd'hui, la crypto-monnaies, le *bitcoin* en particulier qui repose sur l'infrastructure virtuelle « *blockchain* », est devenue un phénomène primordial qui a bouleversé le chemin traditionnel des transactions financières, vers un nouveau processus qui a changé les règles du jeu dans de nombreux secteurs économiques, avec son niveau d'anonymat qu'il offre, qui reste souvent critiqué.

Des nouvelles études, notamment pour les chercheurs de l'université de *Luxembourg*, ont démontrés la faiblesse d'anonymat du réseau *Bitcoin* (absence d'anonymat total) en déterminant l'identification des transactions, même si le client utilise des pseudonymes différents, ce qui mène à la déanonymisation (désanonymisation ou bien déanonymisation) de ces clients. Afin d'y arriver, plusieurs méthodes ont été traitées, en particulier l'apprentissage automatique, en faisant intervenir ses différents modèles, techniques et algorithmes, notamment les réseaux de neurones artificiels.

2 Problématique

Dans le cadre de notre travail, nous essayons de déterminer une approche de désanonymisation des clients Bitcoins, en classifiant les transactions effectuées entre les acteurs du réseaux au tant que « licite »ou « illicite », par rapport aux caractéristiques de ces transactions telles que le nombre d'entrées/sortie pour chaque transaction, le nombre de bitcoins dépensés, le nombre de sortie pour chaque entrée, l'unité de temps où une transaction a

été diffusée sur le réseau Bitcoin, etc.

L'approche se base sur la méthode d'apprentissage automatique la plus connue « l'apprentissage profond »(*Deep Learning*, *Deep Structured Learning*, ou *Hierarchical Learning*), qui regroupe, à son tour, deux types de méthodes couramment utilisées, l'apprentissage supervisé et le non supervisé, qui permettent de manipuler des représentations plus complexes, par son exceptionnelle puissance de calcul avec de grande quantité de données.

3 Contribution

Dans le cadre de ce travail, nous proposons d'exploiter l'alliance des deux technologies « *blokchain* et réseaux de neurones artificiels », en termes de la puissance de calcul, la rigueur et la méthodologie statistiquement précise, en s'appuyant sur une mémoire immense et fiable. Il sera utilisé pour déterminer des meilleures approches pour la classification et la désanonymisation des transactions *Bitcoin* en utilisant une collection de données étiquetées.

En définitive, la contribution du présent travail est de donner des solutions à des problèmes liés à :

- La modélisation des modèles complexes pour l'approche de dé-anonymisation.
- L'association des transactions à des identités réelles.
- L'entrainement, l'optimisation et l'évaluation de ces modèles en testant ces différents paramètres.

4 Organisation du mémoire

Notre mémoire est organisé en deux parties englobant en tout quatres chapitres.

La première partie comporte les deux premiers chapitres. Le premier chapitre présente la notion de l'intelligence artificielle en détaillant l'apprentissage automatique. Nous présentons son fonctionnement global, ces différents types, ainsi que ces différents algorithmes existant. Une attention particulière a été accordée lors de cette étude pour comprendre le fonctionnement des réseaux de neurones artificiels. Le deuxième chapitre est dédié à la présentation de la *blockchain* et la crypto-monnaie virtuelle, notamment le *bitcoin*. Nous

s'intéressons à la cryptographie et la sécurité de cette nouvelle technologie, en citant les différents algorithmes utilisés pour l'assurer.

La deuxième partie comporte les deux chapitres dédie au développement et au déploiement des modèles créés. Le troisième chapitre explique le concept d'anonymat et le dé-anonymat dans le *Bitcoin*. Nous y décrivons les différents modèles crées, ainsi que leurs architectures et les différents paramètres utilisés dans la phase de validation. Le quatrième chapitre est dédié à l'implémentation de nos modèles. Nous commençons le chapitre par la description de l'environnement de développement, le langage, les logiciels et les librairies utilisés. A la fin, nous présentons les résultats obtenus pour chaque modèle proposé, en effectuant une comparaison entre eux afin de tirer le mieux adapté pour la désanonymisation des transactions *bitcoin*.

Apprentissage automatique

1.1 Introduction

L'apprentissage automatique (*machine learning* en anglais) est l'un des champs d'étude de l'intelligence artificielle. Il se consacre au développement d'algorithmes permettant à une machine d'apprendre à partir d'un ensemble de données et d'expériences passées.

Avec l'apprentissage automatique nous sommes en mesure de faire doter une machine d'une aptitude d'apprentissage. Il faudra donc se demander comment une machine peut évaluer les données, sélectionner celle qu'elle souhaite retenir, et comment arriver à les apprendre. Ce chapitre examine les différents algorithmes d'apprentissage automatique qui permettent à une machine d'apprendre.

Dans ce chapitre, nous introduisons les différents concepts de l'apprentissage automatique. Nous commençons par définir l'intelligence artificielle et donner un historique sur l'apparition de l'apprentissage automatique. Ensuite, nous représentons le fonctionnement global de l'apprentissage automatique ainsi que ces différents types. Dans la section qui s'ensuit, nous listons les algorithmes d'apprentissage automatique qui existent. Enfin et avant de conclure, nous revenons sur les défis de l'apprentissage automatique ainsi que les domaines d'application en faisant intervenir l'apprentissage automatique.

1.2 L'intelligence artificielle

1.2.1 Définition

L'intelligence artificielle, souvent abrégée avec le sigle IA, est définit par l'un de ses créateur « Marvin Lee Minskey » comme « la construction de programmes informatiques qui s'adonnent à des tâches qui sont, pour l'instant, accomplies de façon plus satisfaisante par des êtres humains car elles demandent des processus mentaux de haut niveau tels que : l'apprentissage perceptuel, l'organisation de la mémoire et le raisonnement critique » [1].
De manière plus générale, l'intelligence artificielle consiste à mettre en œuvre un certain nombre de techniques visant à permettre aux machines d'initier une forme d'intelligence réelle.

1.2.2 L'IA forte et L'IA fiable

l'intelligence artificielle se présente en deux approches :

- **L'intelligence artificielle forte**
  Fait référence à une machine capable de produire un comportement intelligent, et aussi d'éprouver une impression d'une réelle conscience de soi. La machine comprend donc ce qu'elle fait.

- **L'intelligence artificielle faible**
  Fait référence à une machine qui reproduit un comportement spécifique, mais pas son fonctionnement. En d'autres termes, la machine ne comprend pas ce qu'elle fait.

1.3 Quelques mots d'histoire

Dans le domaine scientifique relativement jeune de l'informatique, l'apprentissage automatique joue un rôle de plus en plus essentiel. Les premières études remontent à des travaux de statistiques dans les années 1920. En 1936 la notion des algorithmes qui apprennent apparait, cette potentialité était évidente dès les travaux de Turing inventant a la même année la machine qui porte son nom [2]. Mais ce n'est qu'après la seconde guerre mondiale que les premières expériences deviennent possible. Se développe ensuite dans les années 1960 les approches connexionnistes avec des perceptrons, et la reconnaissance des formes. La mise en évidence des limites du perceptron simple arrête toutes les recherches dans ce domaine jusqu'à la renaissance dans les années 1980. Les années 1970 sont dominées par des systèmes mettent l'accent sur les connaissances, les systèmes experts. Les limites de tels systèmes se font sentir dans les années 1980 pendant lesquelles a lieu le retour du connexionnisme avec un nouvel algorithme d'apprentissage [3].
A partir des années 1980, l'intelligence artificielle connait un bouleversement. De nombreux scientifiques font des avancés majeurs dans leurs recherches, ce qui mène plusieurs pays à s'investir dans ce domaine.
Dans les années 1990, l'intelligence artificielle a connu des développements fondamentaux concernant, entre autres, l'apprentissage, la planification, la compréhension du langage et sa traduction, la réalité virtuelle, les jeux ou encore l'exploration de données. Cette évolution si rapide est évidemment liée à l'augmentation de la puissance des ordinateurs et des données auxquels les systèmes ont désormais accès.

1.4 Définitions de L'apprentissage automatique

Il existe plusieurs définitions d'apprentissage automatique, nous citons deux parmi eux :

- **Définition 1**
  L'apprentissage automatique correspond au domaine qui se consacre au développement d'algorithmes, permettant à une machine d'apprendre à partir d'un ensemble de données et d'y extraire des concepts et patrons caractérisant ces données [4].

- **Définition 2**
  L'apprentissage automatique est une notion qui englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle. Il existe deux tendances principales en apprentissage : celle de l'intelligence artificielle et qualifiée de symbolique, et celle issue des statistique et qualifiée de numérique [5].

1.5 Fonctionnement général de l'apprentissage automatique

L'apprentissage automatique enseigne aux ordinateurs à faire ce qui est naturel chez un humain. Le modèle d'apprentissage automatique est créé de manière automatique à l'aide d'un algorithme d'apprentissage se servant de méthodes de calcul pour « apprendre » directement à partir des données brutes [6]. En d'autres termes, l'algorithme d'apprentissage analyse les données pour créer le modèle. Une fois créé, il va être exploiter sur des nouvelles données pour faire des prédictions.

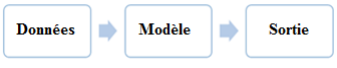


FIGURE 1.1 – Schéma représentant le fonctionnement de l'apprentissage automatique.

1.5.1 Les données (échantillons-observations-exemples)

En ce qui concerne l'apprentissage automatique, la partie la plus importante du processus de formation consiste à acquérir suffisamment de données de qualité pour créer le modèle. Les données doivent être précises et pertinentes pour l'application à développer.
Il y'en existe deux types de données dans l'apprentissage automatique :

- **Les données étiquetées**
  On associe pour chaque donnée une étiquette qui identifie la classe (ou la catégorie) à laquelle elle appartient. On donne par exemple des données étiquetées par leurs sortie, le cas pour prédire les prix d'un immeuble.

- **Les données non-étiquetées**
  Les données ne sont pas accompagnées par une étiquette. Elles sont beaucoup plus accessible malgré la difficulté rencontrée lors de leur exploitation.

1.5.2 Modèle

Un modèle d'apprentissage automatique est une représentation mathématique d'un processus réel. On peut le définir comme une fonction f qui prend des données en entrée, et qui renvoie une sortie.

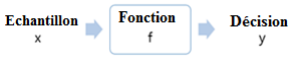


FIGURE 1.2 – Schéma représentant un modèle d'apprentissage automatique.

1.5.3 La sortie

On distingue deux types de sorties :

- **Sortie catégorisée ou discrète**
  Le nombre de valeurs qu'elle prend est fini. Dans ce cas les sorties possibles sont appelées classes.

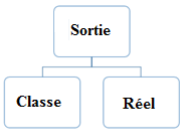


FIGURE 1.3 – Schéma représentant les deux types de sortie.

- **Sortie non-catégorisée ou continue**
  Le nombre de valeurs qu'elle prend est infini. Dans ce cas les sorties sont des valeurs réelles.

1.6 Types d'apprentissage automatique

On distingue plusieurs types d'apprentissage automatique représentés dans la figure suivante :

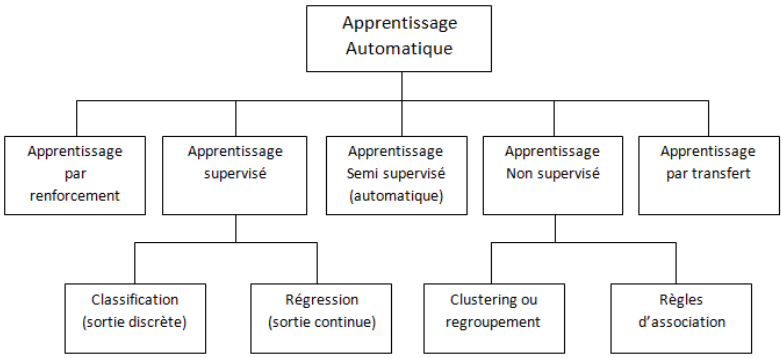


FIGURE 1.4 – Schéma représentant les types d'apprentissage automatique.

1.6.1 Apprentissage supervisé

Ce type d'apprentissage a pour objectif la conception d'un modèle reliant des données d'apprentissage étiquetées bien définies à un ensemble de valeurs de sortie [7]. En d'autres termes, fournir à la machine des données étiquetées X et Y (la donnée X est étiquetée par sa sortie), dont la machine apprend le lien entre la donnée et sa sortie dans la phase d'apprntissage (montrer à la machine le résultat voulu pour lui apprendre). Cet apprentissage peut être

divisé en deux sous-catégories :

- **La classification**
  Lorsque la valeur cible à prédire est discrète.

- **La régression**
  Lorsque la valeur cible est continue grâce à une droite, par exemple si on veut déterminer le prix d'une maison de $700m^2$.

Ce type d'apprentissage automatique est utilisé par exemple dans une application qui est capable de faire la distinction entre plusieurs millions d'animaux, en se basant sur des images déjà classifiées.

1.6.2 Apprentissage non supervisé

Ce type d'apprentissage automatique vise à concevoir un modèle structurant l'information. La différence entre cet apprentissage et le supervisé est dans les catégories ou encore les classes des données d'apprentissage qui sont inconnus (c'est bien le but à atteindre)[7]. On fournit à la machine uniquement les données X et on lui demande d'analyser la structure des données pour apprendre elle même à réaliser certains taches.

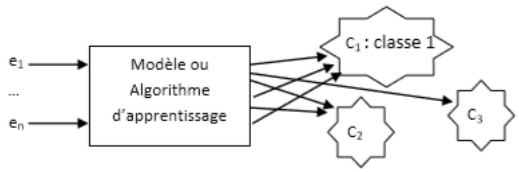


FIGURE 1.5 – Schéma d'un modèle non supervisé.

Cet apprentissage peut être divisé en deux sous-catégories :

- **Le Clustering**
  Classer des données en les regroupant uniquement selon leurs ressemblance. Il est utilisé dans la classification des *Tweets*par exemple.

- **Les règles d'associations**
  Fouiller les données pour extraire des données cachées (items). En d'autres termes, prédire quel événement va se réaliser avec une certaine probabilité. Elle permet de déterminer la relation entre un en-

semble d'items. Ce type est souvent utilisé dans le *Web Mining*.

L'apprentissage non supervisé est utilisé lorsque le problème nécessite une quantité massive de données, par exemple dans les applications des réseaux sociaux telles que Twitter, Instagrame et Snapchat.

1.6.3 Apprentissage semi-supervisé (automatique)

Les données d'entrée sont constituées d'exemples étiquetés et non-étiquetés. Ce qui peut être très utile quand on a deux types de données, car cela permet de ne pas en laisser de côté et d'utiliser toute l'information [7].

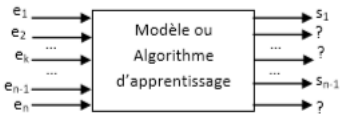


FIGURE 1.6 – Schéma d'un modèle semi-supervisé.

1.6.4 Apprentissage par renforcement

Les données en entrée sont les mêmes que pour l'apprentissage supervisé, cependant cet apprentissage est guidé par l'environnement sous la forme de récompenses, ou de pénalités données en fonction de l'erreur commise lors de l'apprentissage [7]. C'est-à-dire, il faut apprendre un modèle capable de prédire la meilleure décision à prendre étant donné un état de l'environnement. L'algorithme reçoit un retour d'informations de l'analyse des données et il ajuste les paramètres utilisés pour améliorer ses performances. Le système n'est pas formé avec un ensemble de données exemple, mais plutôt, il apprend par le biais d'une méthode d'essais et d'erreurs. En d'autres termes, il consiste à laisser l'algorithme apprend de ses propres erreurs, dont il commence à prédire des décisions aléatoires, s'il trompe, il est pénalisé sinon il est récompensé, et de récompense à une autre il développera sa propre méthode pour accomplir la tache associée.

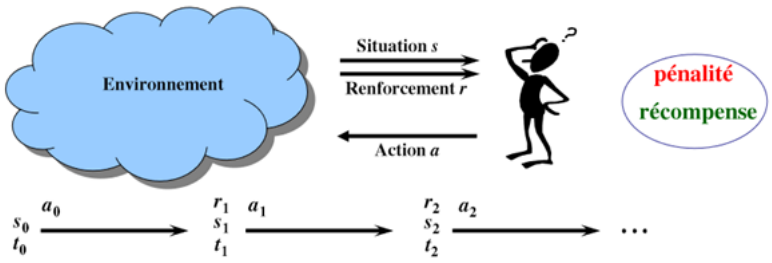


FIGURE 1.7 – Schéma d'un modèle par renforcement.

— L'agent apprend à se rapprocher d'une stratégie comportementale optimale par des interactions répétitives avec l'environnement.
— Les décisions sont prises séquentiellement à des intervalles de temps discrets.

Ce type d'apprentissage automatique est utilisé souvent si on veut apprendre à jouer un jeu, ou apprendre à interagir avec l'environnement.

1.6.5 Apprentissage par transfert

L'apprentissage par transfert consiste à appliquer des connaissances obtenues en effectuant une tâche afin de résoudre un problème différent. Il permet d'exploiter les connaissances tirées d'une tâche source pour améliorer l'apprentissage dans l'exécution d'une nouvelle tâche. Par exemple, les connaissances acquises par un autre algorithme d'apprentissage automatique pour reconnaître des voitures peuvent ensuite être transférées pour être utilisées dans un autre modèle d'apprentissage automatique créé, pour reconnaître d'autres types de véhicules, tels que des camions. C'est-à-dire il est utilisé lorsqu'un programme arrive à exploiter des solutions trouvées pour un problème complexe afin de résoudre un problème moins complexe ou simple.

1.7 Algorithmes d'apprentissage automatique

Un algorithme est un ensemble méthodique d'étapes pouvant servir à effectuer des calculs, à résoudre des problèmes et à prendre des décisions [6]. Plusieurs algorithmes d'apprentissage automatique existent. Le choix du bon algorithme dépend de plusieurs facteurs, notamment la taille, la qualité et la diversité des données. Dans ce qui suit, les algorithmes les plus fréquemment utilisé pour l'apprentissage automatique.

1.7.1 K-plus proches voisins

Dans le courant de l'apprentissage automatique, différents types de classification ont été mis au point, toujours dans le but d'atteindre un degré maximal de précision et d'efficacité, chacun ayant ses avantages et ses inconvénients. Parmi ces types de classification, l'algorithme supervisé K-plus proches voisins qui est nommé par la suite KNN (*K-Nearest Neighbour* en anglais).

1. **Fonctionnement de l'algorithme KNN**
   L'algorithme des plus proches voisins évalue la probabilité selon laquelle une donnée appartient à telle ou telle classe.
   Soit n observations appartenant à E=$\{X_1,X_2,.....,X_n\}$, répartie en m classes $\{C_1,C_2,...,C_m\}$. Chaque donnée i (avec $1 \leq i \leq n$), est caractérisé par une classe C_j avec $(1 \leq j \leq m)$, et un vecteur x_i de dimension p($\vec{x}_i$=($x_1,x_2,.....,x_p$)), qui représente les attributs de la donnée i.
   Soit une donnée X_k qui n'appartient pas à l'ensemble E et qui ne possède pas de classe (X_k est uniquement caractérisé par un vecteur $\vec{x}_k$ de dimension p). Pour trouver la classe de la donnée X_k, l'algorithme KNN :

   — Calculer la distance entre la donnée X_k et chaque donnée X_i (avec $1 \leq i \leq n$) appartenant à E en utilisant la distance euclidienne comme suit :

   $$d(X_k, X_j) = d((x_{k1}, x_{k2}, ..., x_{kp}), (x_{i2}, x_{i2}, ..., x_{ip}))[8]$$

   $$d(X_k, X_j) = \sqrt{(x_{k1} - x_{i1})^2 + (x_{k2} - x_{i2})^2 + ... + (x_{kp} - x_{ip})^2}[8]$$

   avec ($x_{k1},x_{k2},...,x_{kp}$) est le vecteur d'attributs de X_k, et ($x_{i1},x_{i2},...,x_{ip}$) est le vecteur d'attributs de X_i.

   — On retient les k données du jeu données E les plus proches de X_k.
   — On attribut à X_k la classe qui est la plus fréquente parmi les K données les plus proches.

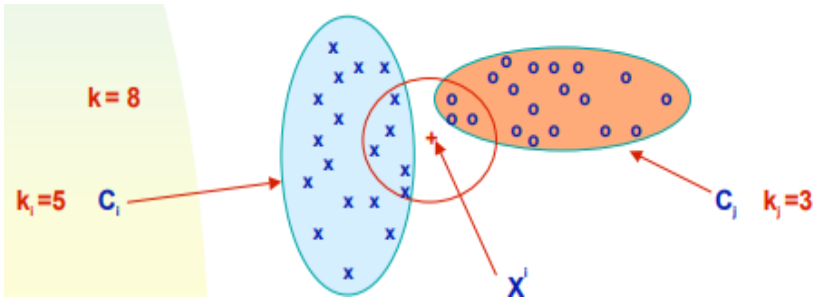


FIGURE 1.8 – Choix de la classe selon les plus proches voisins.

1.7.2 Les arbres de decision

C'est un outil d'aide à la décision ou d'exploitation de données qui permet de représenter un ensemble de choix sous la forme graphique d'un arbre. C'est une méthode d'apprentissage supervisé la plus populaire pour les problèmes de classification des données.

1. **Principe des arbres de décision**
   La technique des arbres de décision est fondée sur l'idée simple de réaliser la classification d'un objet par une suite de test sur les attributs qui le décrivent. L'ensemble des tests possibles est organisé sous forme d'un arbre. Une feuille de cet arbre désigne une des classes(mais à chaque classe peuvent correspondre plusieurs feuilles), et à chaque nœud est associé un test portant sur un ou plusieurs attribut, la réponse à ce test désignera le fils du nœud vers lequel on doit aller. La classification s'effectue donc en partant de la racine pour poursuivre récursivement le processus jusqu'à ce qu'on rencontre une feuille.

2. **Construction récursive d'un arbre de décision**

   - **Cas des attributs binaires**
     On dispose d'un ensemble d'apprentissage X de n données (observations), dont l'un est noté $(Cj,\vec{x}_i)$. Cet exemple est décrit par le vecteur d'attributs $\vec{x}_i$=$(x_1,x_2,...,x_p)$, et par une classe C_j, avec j varie entre 1 et m.
     on suppose que les attributs sont une valeur binaire(soit vrai, soit faux).
     L'algorithme récursive de construction d'un arbre de décision est :
     Début
       Si tout les données de X appartiennent à la même classe alors

Créer une feuille portant le nom de cette classe.
Sinon
Choisir le meilleur attribut pour créer un nœud.
Le test associe à ce nœud sépare X en deux partie notée X_g et X_d (sous-arbre gauche et sous-arbre droit).
Construire-arbre (X_g)
Construire-arbre(X_d)
Fin [5].

- **Cas des attributs non binaires**

  — **Le cas nominal**
  les cas ou les attributs sont une valeurs discrètes se généralise facilement quand le test que l'on construit se réduit à opposer une valeur à toutes les autres : on est alors ramené au cas binaire. Par exemple, s'il existe un attribut couleur prenant ses valeurs dans l'ensemble {bleu, rouge,vert,jaune}, il est simple de l'éclater en quatre attribut binaire, tel couleur_bleu qui est vrai ou faux sur chaque donnée d'apprentissage.

  — **Le cas continue**
  Dans ce cas le test consistera à comparer les valeur à un seuil pour construire un nœud binaire.

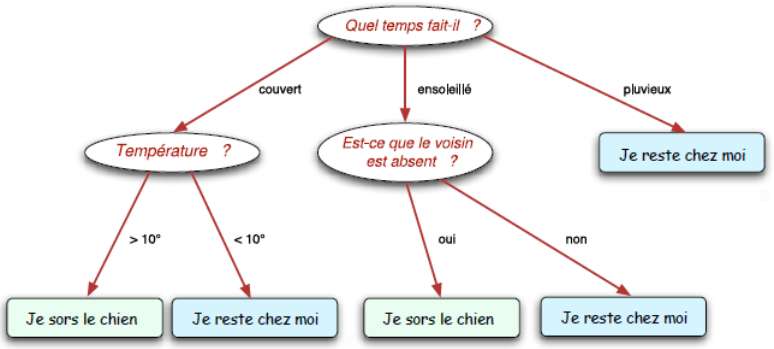


FIGURE 1.9 – Choix de la classe selon les plus proches voisins.

1.7.3 Classification naive de Bayes

Naive Bayes est un classificateur probabiliste d'apprentissage automatique supervisé, simple mais efficace. Il est particulièrement utile pour les problématiques de classification de texte.

1. **Fonctionnement de classification naive de bayes**
   Le but de tout classificateur probabiliste est, avec les caractéristiques (attributs) x_{i1},x_{i2},...,x_{ip} et les classes C_1 à C_m, de déterminer la probabilité que les caractéristiques se produisent dans chaque classe et de renvoyer la classe la plus probable. Par conséquent, pour chaque classe il faut calculer $P(c_j|x_{i1}, x_{i2}, ..., x_{ip})$.
   Pour ce faire, nous utilisons le théorème de Bayes, qui nous dit :

   $$P(A|B) = \frac{P(B|A)*P(A)}{P(B)}[9]$$

   Afin de simplifier le calcule, on suppose que les attributs x_{i1},x_{i2},...,x_{ip} sont conditionnellement indépendants donc :

   $$P(c_j|x_{i1}, x_{i2}, ..., x_{ip}) = \frac{P(x_{i1},x_{i2},...,x_{ip}|c_j)*P(c_j)}{P(x_{i1})*P(x_{i2})*...P(x_{ip})}[9]$$

   avec

   $$P(x_{i1}, x_{i2}, ..., x_{ip}|c_j) = P(x_{i1}|C_j)*P(x_{i2}|C_j)*...P(x_{ip}|C_j)[9]$$

   Par exemple, si on a un jeu de données sur 1000 fruits, on dispose de 3 classes : banane, orange et autre, avec les attributs : long, sucré, jaune.
   Supposons que quelqu'un nous demande de lui donner le type d'un fruit qu'il a, avec les attributs suivants :
   il est jaune, il est long, il est sucré
   On calcule les probabilité :
   p(banane|long,jaune,sucre)=0.969
   p(orange|long,jaune,sucre)=0
   p(autre|long,jaune,sucre)=0.072
   On remarque que la probabilité que notre fruit soit une banane est largement plus grande que celle des autres probabilités. On classifie notre fruit inconnu comme étant une banane.

1.7.4 Algorithme K-moyenne

K-moyenne (*K-means* en anglais), est un algorithme non supervisé de clustering. K-moyenne permet de regrouper en K clusters distincts les observations, de façon est ce que les données similaires se retrouveront dans un même cluster. Par ailleurs, une observation ne peut se retrouver que dans un cluster à la fois(exclusivité d'appartenance)

1. **Notion de similarité**
   En apprentissage non supervisé les données sont représenté sous forme d'une matrice, comme suit : $X = \begin{pmatrix} x_1 1, & x_1 2, & ..., & x_1 n \\ x_2 1, & x_2 2, & ..., & x_2 n \\ . & & & \\ . & & & \\ x_m 1, & x_m 2, & ..., & x_m n \end{pmatrix}$

   Chaque ligne représente un individu (observation), à l'issue de l'application du clustering, on retrouve ces données regroupées par ressemblances [10].
   Pour pouvoir regrouper un jeu de données en K clusters distincts, l'algorithme k-moyenne a besoin d'un moyen de comparer le degré de similarité entre les différentes observations. Ainsi, deux données qui se ressemblent auront une distance de dissimilarité réduite, alors que deux données différentes auront une distance de séparation plus grande.
   La définition de distance, la plus connues pour le cas de clustering est la distance Euclidienne, qui, pour deux observation X_i et X_j, se calcule comme suit :

$$\begin{aligned} d(X_i, X_j) &= d((x_{i1}, x_{i2}, ..., x_{ip}), (x_{j2}, x_{j2}, ..., x_{jp}))[10] \\ &= \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{ip} - x_{jp})^2}[10] \end{aligned}$$

   Avec $(x_{i1}, x_{i2}, ..., x_{ip})$ est le vecteur des attributs de X_i, et $(x_{j2}, x_{j2}, ..., x_{jp})$ le vecteur des attributs de X_j.

2. **Choisir K : le nombre de clusters**
   Choisir un nombre de clusters K n'est pas forcément intuitif. Du fait qu'il n'existe pas de procédé automatique pour trouver le bon nombre de clusters, la méthode la plus usuelle est de lancer K-moyenne avec

différentes valeurs de K, et de calculer la variance des différents clusters. La variance est la somme des distances entre chaque centre d'un cluster (centroid) et les différentes observations inclues dans le même cluster, qui se calcule comme suit :

$$V = \sum_{j} \sum_{j}^{C_j} D(C_j, X_i)^2 [10]$$

Avec
C_j : représente le centroid du cluster.
X_i : la i ème observation dans le cluster ayant pour centroid C_j.
$D(C_j, X_i)$: la distance euclidienne entre le centre du cluster et l'observation X_i [10].
Donc, on cherche à trouver un nombre de clusters K de telle sorte que les clusters retenus minimisent la distance entre leurs centres (centroids), et les observations du même clusters. On parle de minimisation de la distance (inertie) intra-classe.

1.7.5 Les algorithmes génétiques

Les algorithmes génétique sont des algorithmes non supervisé qui n'ont pas besoin d'exemples pour apprendre, et aucune base n'est nécessaire pour l'apprentissage. En effet, ces algorithmes ne sont pas un outil, mais avant tout une méthode qui permet d'obtenir des solutions ultra-performantes à partir de rien, pour résoudre des problèmes qui n'ont pas une solution unique ou pas exacte en un temps raisonnable.

1. **les notions des algorithmes génétique**
   Dans ce type d'algorithmes nous retrouvons les notions de population, d'individu, de chromosome et de Gène avec trois opérateurs d'évolutions, l'opérateur de sélection, l'opérateur de croisement et l'opérateur de mutation définit comme suit [11] :
   - **La population :** est l'ensemble des solutions envisageables.
   - **L'individu :** représente une solution.
   - **Le chromosome** est une composante de la solution.
   - **Le Gène :** est une caractéristique, une particularité.
   - **La sélection :** choix des individus les mieux adaptés.
   - **Le croisement :** reproduction par mélange des particularités des individus choisis.
   - **La mutation :** altération aléatoire des particularités d'un individu.

2. **Fonctionnement des algorithmes génétique**

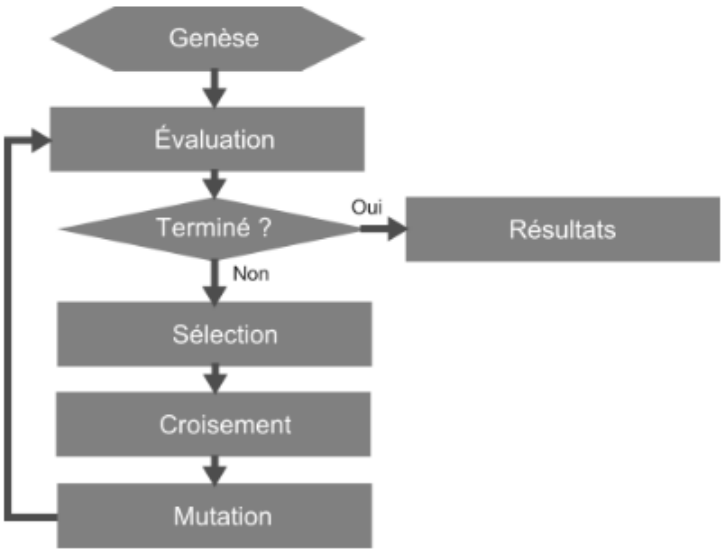


FIGURE 1.10 – Fonctionnement des algorithmes génétiques

La genèse est l'étape de la création d'une population aléatoire. C'est le point de départ de notre algorithme. L'évaluation est l'analyse si une solution est optimale. Nous effectuons une boucle tant que l'évaluation estime que la solution n'est pas optimale [11].

1.7.6 Les réseaux de neurones artificiels

1. **Réseaux de neurones**
   Un réseau de neurones est l'association, en graphe plus ou moins complexe, d'objets élémentaires, les neurones formels. Plus précisément, est un ensemble de neurones virtuels disposés en un réseau virtuel dont chaque neurone reçoit de l'information entrante et émet de l'information sortante. Un réseau de neurones est entrainé grâce à un mécanisme d'apprentissage (n'est pas programmable), utilisé en apprentissage automatique pour construire un modèle à partir de données existantes, dans le but d'exécuter des prédictions sur de nouvelles données, soit à l'aide de la régression, dans le cas de données continues, ou de classification, dans le cas de données discrètes [12].

2. **Les réseaux de neurones artificiels et les réseaux de neurones profond**
   Les réseaux de neurones artificiels sont un système d'apprentissage supervisé constitué d'un grand nombre d'éléments simple appelés neurones ou perceptrons. Chaque neurone peut prendre des décisions simples et alimente ces décisions vers d'autres neurones, organisés en couches inter-connectées. Le réseau de neurones peut émuler presque toutes les fonctions et répondre pratiquement à toutes les questions, avec suffisamment d'échantillons de formation et de puissance de calcul. Un réseau neuronal "peu profond" ne comporte que trois couches de neurones :
   — Une couche d'entrée qui accepte les variables, ou entrée indépendante du modèle
   — Une couche cachée
   — Une couche de sortie qui génère des prédictions

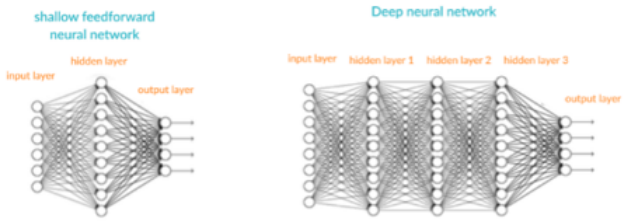


FIGURE 1.11 – Réseaux de neurones artificiel vs Réseaux de neurones profond.

   Un réseau neuronal profond a une structure similaire, mais il a deux ou plusieurs « couches cachées » de neurones qui traitent les entrées pour améliorer la précision des résultats obtenus. Aujourd'hui, la plupart des modèles et implémentations de réseaux neuronaux utilisent un réseau profond entre 3 à 10 de couches de neurones.

3. **Concepts de réseaux de neurones artificiels**

   - **Entrées**
     Données sources (généralement un ensemble de valeurs réelles) alimentées dans le réseau neuronal, dans le but de prendre une décision, ou une prédiction sur les données.

- **Ensemble d'entraînement**
  Un ensemble d'entrées pour lesquelles les sorties correctes sont connues, utilisées pour entrainer le réseau neuronal.

- **Les sorties**
  Un ensemble de valeurs réelles ou de décisions booléennes dont chaque valeur de sortie est générée par l'un des neurones de la couche de sortie.

- **Neurone/ Perceptron**
  L'unité de base du réseau neuronal. Accepte une entrée et génère une prédiction à l'aide d'une fonction d'activation.

- **Poids**
  Une valeur numérique qui définit la sortie de chaque neurone, il est ajouté lors de la contribution des signaux pour déterminer l'activation ou la désactivation du neurone. Il sert à propager l'information (on fait varier les poids pour minimiser les erreurs, c'est-à-dire on cherche la valeur du poids qui donnera le meilleur résultat).

- **Passe avant**
  le passage de l'information d'une couche à une autre couche jusqu'à obtenir le résultat final.

- **Fonction d'erreur**
  Définit la distance entre la sortie réelle du modèle actuel et la sortie correcte. Lors de la formation du modèle, l'objectif est de minimiser la fonction d'erreur et de rapprocher le plus possible la sortie de la valeur correcte.

- **Biais et variance**
  Le biais mesure l'adéquation du modèle avec l'ensemble de formation, capable de prédire correctement les résultats connus des exemples de formation.
  La variance mesure l'efficacité du modèle avec des entrées inconnues qui n'étaient pas disponibles pendant la formation.

- **Hyperparamètres**
  Un hyperparamètre est un paramètre qui affecte la structure ou le fonctionnement du réseau neuronal. Dans de vrai projets d'apprentissage en profondeur, le réglage des hyperparamètres est le

principal moyen de construire un réseau qui fournit des prédictions précises pour un certain problème. Les hyperparamétres courants incluent le nombre de couche cachées, la fonction d'activation et le nombre de répétitions (d'époques) d'entrainement.

4. **Fonctionnement général**
   Un réseau de neurones est un assemblage de constituant élémentaires inter-connectés dont chaque neurone fonctionne indépendamment des autres de telle sorte que l'ensemble forme un système massivement parallèle. Dans cette structure, les neurones de la couche d'entrée captent les données pour les envoyer sous forme de signaux pondérés par des poids aux neurones de la couche suivante (si la donnée correspond à son activation). Les dendrites reçoivent des signaux, la fréquence de chaque signal dépend du poids « W_i ». Ces signaux sont ensuite transmis vers le corps de la cellule. Cette dernière est caractérisée par deux éléments importants : la fonction d'activation et la condition d'activation. C'est en vérifiant la condition d'activation qu'on active l'axone en ajoutant un biais et en appliquant une fonction d'activation aux signaux reçus, pour les émettre à nouveau aux neurones qui suivent à travers leurs dendrites et ainsi de suite.

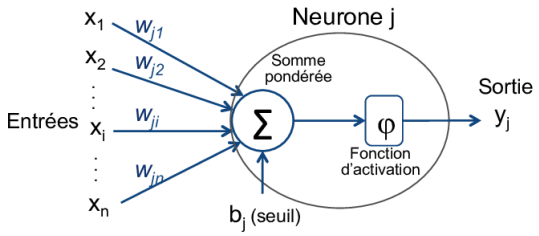


FIGURE 1.12 – Fonctionnement Un réseau de neurones artificiel.

5. **Le neurone formel**

   - **Modélisation d'un neurones formel**
     Un neurone formel (ou simplement neurone) est une fonction algébrique non linéaire, bornée, dont la valeur dépend des paramètres appelés poids. Les variables de cette fonction sont habituellement appelées « entrée » du neurones, et la valeur de la fonction résultante est appelées sa « sortie » [13].
     — Les x_i représente les vecteurs d'entrée (données d'entrées), elles proviennent soit des sorties d'autres neurones, soit de stimuli sensoriels (capteur visuel, sonore,..)

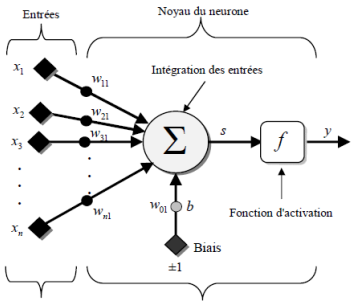


FIGURE 1.13 – Modèle d'un neurone artificiel.

— Les w_{ij} sont les poids synaptiques du neurone. Ces poids pondèrent les entrées et peuvent être modifiés par apprentissage.

— Biais : entrée prend souvent les valeurs -1 ou +1 qui permet d'ajouter de la flexibilité au réseau en permettant de varier le seuil de déclenchement du neurone par l'ajustement des poids et du biais lors de l'apprentissage.

— Noyau : intègre toutes les entrées et le biais et calcul la sortie du neurone selon une fonction d'activation.

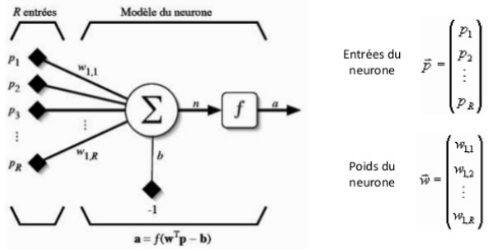


FIGURE 1.14 – Modèle mathématique d'un neurone artificiel.

Un neurone est essentiellement constitué d'un intégrateur qui effectue la somme pondérée de ses entrées. Le résultat s de cette somme est ensuite transformé par une fonction d'activation (appelée aussi fonction de transfert) f qui produit la sortie y du neurone. Les différentes étapes d'apprentissage du perceptron se résument comme suit :

— Prend les entrées qui sont introduite dans les perceptrons dans la couche d'entrée, les multiplie par leurs poids et calcule la somme.
— Ajoute le numéro un (1), multiplié par un « poids de biais ». Il s'agit d'une étape technique qui permet de déplacer la fonction de sortie de chaque perceptron (la fonction d'activation) vers le haut, le bas, la gauche et la droite sur le graphique numérique.
— Alimente la somme par la fonction d'activation dans un système de perceptron simple.
— Le résultat de la fonction est la sortie.

6. **Architecture des réseaux de neurones**
L'architecture d'un réseau de neurones est l'organisation des neurones entre eux au sien d'un même réseau, qui dépend de la tache à apprendre.
Selon la topologie de connexion des neurones, on peut classer les réseaux de neurones en deux grandes catégories :

— Les réseaux de neurones statiques (acycliques ou non bouclés - *Feedforword*).
— Les réseaux de neurones dynamiques (récurrents ou bouclés - *Feedback*)

- **Les réseaux de neurones non bouclés**
Un réseau de neurones non bouclé est représenté graphiquement par un ensemble de neurones « connectés » entre eux, dont l'information circule des entrées vers les sorties sans retour en arrière. Ce type de réseau est utilisé généralement pour effectuer des taches de classification, ou de modélisation des processus statiques non linéaires [14].
Il existe 3 types de réseaux de neurones non bouclés.

— **Les réseaux de neurones mono-couche**
Un réseau simple composé d'une seule couche d'entrée entièrement connectés à une seule couche de sortie, conçu dans le but de la reconnaissance des formes [15].

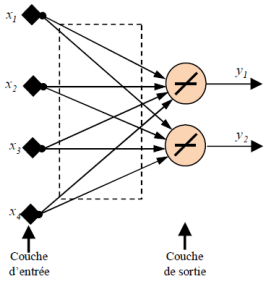


FIGURE 1.15 – Schéma d'un réseau de neurones mono-couche.

— **Les réseaux de neurones multi-couches**
Les neurones sont arrangés par couches. Il n'y a pas de connexions entre les neurones d'une même couche, et les connexions ne se font qu'avec les neurones de couches avales (en sortie).
Il existe trois type de couches :
La couche d'entrée (*Input layer*) : est la couche à laquelle on transmet les fonctionnalités du jeu de données. Elle sert à transmettre les entités de données aux couches cachées.
Les couches cachées (*hidden layers*) : sont les couches entre la couche d'entrée et la couche de sortie, qui effectuent les calcules et transmettent les informations à la couche en sortie.
La couche de sortie (*Output layer*) : est la couche qui donnera les résultats après la formation du réseau.
Les réseaux multi-couches sont beaucoup plus puissants que les réseaux simples à une seule couche.

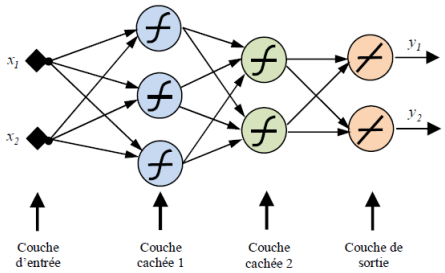


FIGURE 1.16 – Schéma d'un réseau de neurones multi-couches.

— **Les réseaux de neurones à connexion locales**
Une structure multi-couches,dont chaque neurone entretient des relations avec un nombre réduit et localisé de neurones

de la couche avale. Les connexions sont donc moins nombreuse que dans le cas d'un réseau multi-couches classique [15].

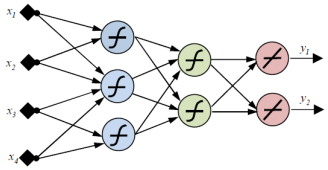


FIGURE 1.17 – Schéma d'un réseau de neurones à connexions locales.

- **Les réseaux de neurones bouclés**
  Contrairement aux réseaux de neurones non bouclés, les réseaux de neurones bouclés sont des système dynamique, construis par des équations différentielles qui peuvent avoir une topologie de connexion quelconque, comprenant des boucles qui ramènent aux entrées la valeur d'une ou plusieurs sorties [15].

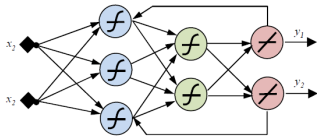


FIGURE 1.18 – Schéma de réseau de neurones bouclé.

7. **Les fonctions d'activation du réseau neuronal**
   La fonction d'activation est une équation mathématique qui prend l'entrée de chaque neurone et la transforme en sortie, généralement entre « 1 » et « 0 » ou entre « -1 » et « 1 ». Elle détermine la sortie du modèle, son efficacité du calcul et sa capacité à s'entrainer et à converger après plusieurs itérations de formation [16].
   Les fonctions d'activations les plus utilisées sont : Sigmoid, ReLu, Swish, la fonction linéaire, la fonction seuil.

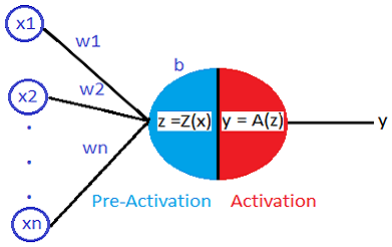


FIGURE 1.19 – Schéma représentant la fonction d'activation.

8. **La rétro-propagation dans les réseaux de neurones**
   Un réseau de neurones est défini avec des poids initiaux pour générer la prédiction initiale. Pour définir la distance entre le modèle et la vrai prédiction, une fonction d'erreur est utilisée, calculée en vérifiant à quelle distance la prédiction est éloignée de la vraie valeur connue. La rétro-propagation est un algorithme qui peut découvrir les poids optimaux relativement, rapidement même pour un réseau avec des millions de poids, dans le but de minimiser et générer la plus petit erreur [17].

   - **La rétro-propagation dans le monde réel**
     Dans le monde réel, on ne peut coder probablement pas une implémentation de rétro-propagation, car d'autres chercheurs l'ont déjà fait pour faciliter le travail, c'est ce qu'on appelle les *frameworks*. Dans les réseaux de neurones, on peut travailler avec des *frameworks* d'apprentissage en profondeur comme *« Tensorflow »* et *« Keras »*, qui contiennent des implémentations efficaces de rétro-propagation, qu'on peut exécuter avec seulement quelques lignes de code.

9. **Biais en réseaux de neurones**
   Dans les réseaux de neurones artificiels, le mot biais a deux significations :
   — Un neurone de polarisation qui fait partie de la structure du réseau de neuronal
   — Un biais en tant que concept statistique, qui reflète la capacité du réseau à générer des prévisions basées sur les échantillons de formation que l'on fournit.

   Dans chaque couche du réseau neuronal, un neurone biais est ajouté pour stocker une valeur de « 1 ». Le neurone de polarisation permet de déplacer la fonction d'activation sur le graphe numérique. Sans

neurone de polarisation, chaque neurone prend l'entrée et la multiplie par son poids, sans rien ajouter à l'équation d'activation. Cela signifie qu'il n'est pas possible de saisir une valeur de « 0 » et de générer une sortie de « 2 ». Dans de nombreux cas, il est nécessaire de déplacer toute la fonction d'activation pour générer les valeurs de sortie requises, cependant, le neurone de polarisation rend cela possible.

- **Le biais et la variance dans les réseaux de neurones**
  Le biais reflète l'adéquation du modèle avec l'ensemble d'entrainement. Un biais élevé signifie que le réseau neuronal n'est pas en mesure de générer des prédictions correctes, même pour les exemples sur lesquels il s'est entrainé.
  La variance reflète la façon dont le modèle s'intègre aux exemples dans l'ensemble de validation. Une variance élevée signifie que le réseau neuronal n'est pas en mesure de prédire correctement pour de nouveaux exemples qu'il n'a jamais vu auparavant.

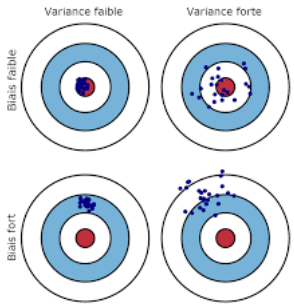


FIGURE 1.20 – Le biais et la variance dans les réseaux de neurones.

10. **Sur-ajustement et sous-ajustement dans les réseaux de neurones**
    - **Le sur-ajustement**
      Se produit lorsque le réseau neuronal est bon pour apprendre son ensemble d'entrainement, mais n'est pas en mesure de généraliser ses prédictions à des exemples supplémentaires. Cela se caractérise par un biais faible et une variance élevée [18].

    - **Le sous-ajustement**
      Se produit lorsque le réseau neuronal n'est pas en mesure de prédire avec précision pour l'ensemble d'apprentissage, sans parler de l'ensemble de validation. Cela se caractérise par un biais élevé et une variance élevée [18].

11. **Hyperparamètres de réseau neuronal**
Les hyperparamètres déterminent la structure du réseau neuronal, sa formation et le fonctionnement de ses différents éléments.

- **La différence entre le paramètre du modèle et l'hyperparamètre :**

  — **Un paramètre de modèle**
  Est un paramètre interne, utilisé pour apprendre et faire des prédictions dans un modèle d'apprentissage en profondeur de production.

  — **Un hyperparamètre**
  Est un paramètre externe défini par l'opération du réseau neuronal tel que le nombre d'itérations d'entrainement, le nombre de couches cachées ou la fonction d'activation.
- **Optimisation hyper-paramétrique dans le monde réel**
Dans un véritable projet de réseau neuronal, on peut soit optimiser manuellement les valeurs des hyperparamètres, en utilisant des techniques d'optimisation dans le cadre d'apprentissage en profondeur, ou en utilisant l'un des nombreux outils d'optimisation hyper-paramètrique tiers. Le *framework « Keras »* par exemple offre des bibliothèques pour l'optimisation des hyperparamètres telle que : *« Hyperopt »*, *« Kopt »* et *« Talos »*. Tandis qu'avec *« Tensorflow »* on peut utiliser *« GPflowOpt »* pour l'optimisation bayésienne et des solutions commerciales comme le *« Cloud Machine Learning Engine »* de *« Google »* qui offrent plusieurs options d'optimisation.

1.8 Les défis de l'apprentissage automatique

L'apprentissage automatique a révolutionné le monde aujourd'hui avec tout ce qu'il propose comme solutions aux différents problèmes actuels. Cela n'élimine pas la présence de quelques défis qui entrave l'utilisation de cette technologie, dont on peut citer l'infection des facteurs externes sur les jeux de données du monde réel utilisés, ce qui les rend imparfaits (inexactes). Le déploiement de grands sous ensemble de données, et de différents algorithmes de classification et de la régression qui augmente la complexité du modèle ce qui mène à l'im-performance. Les données sont la matière première de tout projet d'apprentissage automatique. La collecte et le stockage de ces

données coûte en terme d'argent et du temps (l'apprentissage et les testes pour former la version finale du modèle). Parfois, un projet nécessite des flux de données différents, de plusieurs sources différentes, et pour le faire fonctionner, il faudra fusionner tous ces types de données, ce qui est un peu difficile à réaliser et qui, une fois fait, ne génère pas des résultats précis.

1.9 Domaines d'application de l'AA

Les systèmes d'apprentissage automatique sont utilisés dans de très nombreux domaines de notre vie, dont on peut citer :

- **La reconnaissance de formes (images, vidéos, signaux)**
  Aujourd'hui la classification et la régression de plusieurs formes se fait grâce aux algorithmes supervisés qui font apprendre un modèle comment les reconnaitre.
- **La robotique**
  Les robots sont dotés par des systèmes d'apprentissage automatique qui permettent de maitriser plusieurs taches pour automatiser et faciliter les étapes de travail dans plusieurs domaines tels que la production.
- ***Le trafic***
  Le domaine de transport est doté par des systèmes intelligents leurs permet de manœuvrer indépendamment, sans intervention humain, le trafic réel et favoriser au quotidien la qualité de service, le confort des usagers et la sécurité.
- **Internet**
  Les algorithmes d'apprentissage automatique permet d'effectuer plusieurs taches sur internet pour protéger l'utilisateur (filtres anti spam, défense contre les virus et les malwares) et suivre son mouvement, avec les algorithmes de classement des moteurs par exemple (*RankBrain* de *Google*) pour faciliter la navigation sur internet.
- **Jeux**
  Les développeurs de jeux informatiques actuels utilisent l'apprentissage automatique pour rendre leurs jeux plus intéressants et créer le *« gameplay »* le plus équilibré possible en s'assurant que les adversaires de l'ordinateur s'adaptent de manière intelligente aux comportements des joueurs humains.
- **Science**
  Plusieurs domaines de science qui intègrent l'apprentissage automatique aujourd'hui, citons par exemple le marketing qui a basculé vers le marketing digital, la physique pour enregistrer et traiter les données

de mesure et détecter ainsi les écarts. On cite aussi la médecine qui utilise ces algorithmes pour le diagnostic et le traitement des patients.

1.10 Conclusion

Dans ce chapitre, nous avons introduit les différents notions concernant l'apprentissage automatique. Nous avons mis l'accent sur son fonctionnement global, et les différentes approches adaptées dans ce domaine. Ensuite, nous avons présenté les différents techniques utilisées dans l'apprentissage automatique.Terminions enfin par les domaines d'applications et les défis de l'apprentissage automatique.

Ces informations sont nécessaires pour pouvoir présenter la contribution principale dans notre mémoire, à savoir, la désanonymisation des clients sur le réseau bitcoin. Toutefois, avant d'aller plus loin, nous avons aussi besoin de présenter la technologie du « blockchain » et plus particulièrement la notion de « bitcoin », ce que nous faisons dans le chapitre suivant.

Blockchain bitcoin

2.1 Introduction

La monnaie ou plus l'argent est souvent perçu comme un élément clé dans la relation entre les membres d'une société. Un mode de paiement qui, non seulement, facilite les échanges commerciaux et les différentes transactions entre plusieurs entités, mais aussi nécessite le passage par plusieurs tiers, qui doivent avoir notre confiance pour réussir ces transactions, en devenant ainsi totalement surveillé et contrôlé par l'état.

Cette topologie, avec la digitalisation exponentielle et massive de toutes les activités humaines, rend ces échanges plus couteux, chronophages et risqués pour les particuliers, les investisseurs et les institutions financières, d'où vient le besoin de créer de nouveaux moyens d'échanges, plus rapides et plus efficaces que les méthodes traditionnelles. Cet effet a donné naissance à la crypto-monnaie ou la monnaie virtuelle électronique, connu aussi sous le nom « e-cash ».

Pour rendre la création d'une monnaie virtuelle possible, « la blockchain » a su résoudre le problème de transfert de titre de propriété sans intervention d'un tier de confiance.

Beaucoup de questions méritent d'être posées au sujet de cette technologie. On doit se demander comment elle fonctionne ? Si différents types existent ? Quels en sont les différents avantages et défauts ?

Ce chapitre donne quelques pistes afin de répondre à ces questions. Dans les sections qui suivent, nous présentons le système de la blockchain et la crypto-monnaie virtuelle, à voir le bitcoin qui est le plus connu et le plus utilisé. Nous verrons qu'est ce qu'une blockchain, un bitcoin, ces types et ces caractéristiques essentiels. Nous entamons ensuite la section crypto-graphique, la où nous nous intéressons à la sécurité dans cette nouvelle technologie, et les différents algorithmes utilisés pour garantir la fiabilité et l'intégrité des données et assurer la satisfaction de l'utilisateur. Enfin, nous terminons par présenter les différents domaines d'utilisation de la blockchain et son avenir.

2.2 Blockchain, c'est quoi ?

Une blockchain désigne une chaine de blocs sur lesquels sont stockés des informations de toutes natures : transactions, contrats, titres de propriétés, œuvres, etc. L'ensemble de ces blocs forme une base de données semblable aux pages d'un grand livre de compte. Ce livre est décentralisé, c'est-à-dire qu'il n'est pas hébergé par un serveur unique, mais par une partie des utilisateurs. Les informations contenues sur les blocs sont protégées par plusieurs procédés cryptographiques innovants, ce qui rend leur modification difficile voir impossible [19].

FIGURE 2.1 – Représentation d'une chaîne de blocs.

2.3 Bitcoin, c'est quoi ?

Le bitcoin est la première monnaie virtuelle, monnaie électronique, crée par « Satoshi Nakamoto » [1], déclenchant avec une véritable révolution dans le monde des transactions financières. Il utilise la cryptographie pour valider les transactions entre les acteurs du réseau, permettant d'acheter des biens et des services sur Internet (faire des échanges sur Internet). Son nom « bitcoin » est la concatination des mots « bit » qui correspond à l'unité de mesure binaire, et « coin » qui signifie « pièce de monnaie ». En effet, il n'y a pas besoin de « compte » bitcoin, on possède des bitcoins comme on possède des pièces de monnaie dans son portefeuille, la seule différence c'est que le « bitcoin » n'a pas de banque centrale ni aucun organisme centrale ou institution financière pour le réguler, mais elle repose sur un vaste réseau de pair-à-pair sur Internet. Comme il est important de faire la différence entre « bitcoin » qui correspond à la monnaie virtuelle, et « Bitcoin » qui correspond au protocole associé à la blockchain définissant les lois et le fonctionnement du « bitcoin » [20].

1. Jusqu'aux aujourd'hui les recherches n'ont pas réussi à déterminer l'identité réelle de ce créateur mais l'idée la plus probable c'est que le fondateur de cette technologie n'est pas une seule personne mais plutôt un groupe de chercheurs.

2.4 Quelque mots d'histoire

La blockchain a été conceptualisée et utilisée pour la première fois par un inconnu sous le pseudonyme « Satoshi Nakamoto » pour créer la fameuse crypto-monnaie bitcoin (la blockchain est l'infrastructure virtuelle sur laquelle repose le bitcoin), en 2008. Satoshi s'est appuyé pour cela sur plusieurs publications scientifiques traitant des chaines cryptographiques publiées entre 1991 et les années 2000 [21].
L'historique de bitcoin a été mouvementé depuis le lancement des premiers bitcoins en 2009 et la première véritable transaction effectuée en mai 2010. Début 2011, le bitcoin touche la parité avec le dollar et atteint plusieurs millions de dollars de capitalisation, les premiers articles sur le Bitcoin commencent alors à apparaître dans des journaux majeurs aux Etats-Unis. A partir de 2015, la blockchain a commencé à susciter une grande attention [22]. Aujourd'hui la blockchain est considérée comme une avancée technologique semblable à internet dans les années 1990.

2.5 Types de blockchain

Plusieurs types de blockchain existent. Ces différentes blockchains sont expliquées plus en détail ci-dessous :

2.5.1 Blockchain publique

Il s'agit d'une blockchain accessible à n'importe qui dans le monde. Dont chaque personne ayant accès à internet pourra se connecter sur une plateforme blockchain pour devenir un nœud autorisé dans son réseau, là où il pourra effectuer des transactions. Bitcoin et Ethereum constituent les deux principales blockchains publiques, mais on peut citer aussi Litecoin et Dogecoin [23].

2.5.2 Blockchain privée

Une blockchain privée est une blockchain restrictive (fonctionne dans un réseau fermé), utilisée au sein d'une organisation ou d'entreprises privées pour des cas d'utilisations internes, où seuls des membres sélectionnés sont autorisés à acceder. Exemples de blockchains privées : Multichain, Hyperledger Fabric, Hyperledger Sawtooth, Corda [23].

2.5.3 Blockchain de consortiums

Une blockchain de consortium ou blockchain fédérée est un type semi-décentralisé gérée par plus d'une organisation.
Exemples de blockchains de consortium : Energy Web Foundation, IBM Food Trust [24].

2.5.4 Blockchain hybride

Une blockchain hybride est une combinaison de la blockchain privée et publique, où les utilisateurs peuvent contrôler qui a accès à quelles données stockées dans la blockchain (seule une section sélectionnée de données ou d'enregistrements de la blockchain peut être autorisé à devenir publique en gardant le reste confidentiel), conçu dans le but de déployer le meilleur des deux mondes : publique et privé.
Exemples de blockchains hybrides : Dragonchain, XinFin

2.6 Les caractéristiques essentielles d'une blockchain

Les principes sur lesquels est fondée la blockchain sont les suivants :

- **Une base de donnée distribuée**
  Contrairement aux bases de données plus classiques, les données d'une blockchain ne sont pas hébergées par un serveur unique, mais distribuées entre les utilisateurs du réseau sans aucune autorité centrale de contrôle.

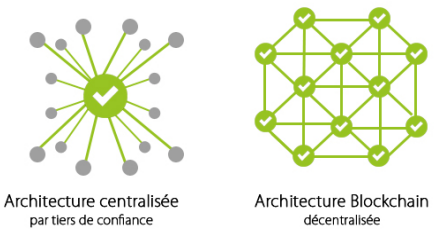


FIGURE 2.2 – Architecture centralisée vs l'architecture blockchain.

- **Transparente**
  La blockchain est accessible par tous les nœuds du réseau, dont chacun peut consulter l'ensemble des échanges inscrit.

- **Sécurisée**
  Chaque modification apportée à la blockchain doit être approuvée et vérifiée par une communauté de paires. Cette communauté est chargée d'empêcher toute tentative de fraude. De ce fait une tentative de falsifier le registre sera immédiatement repérée et invalidée.

- **La transmission d'informations sans organe de contrôle**
  La blockchain est fondée sur des échanges de pair à pair (P2P), en utilisant le principe de signature électronique, afin d'authentifier l'auteur d'une transaction. Une transaction est donc émis sans intermédiaire : celle-ci n'est pas vérifiée par un serveur centrale, mais par les nœuds du réseau.

- **Mineurs**
  Un mineur est un individu vérifiant les transactions et les opérations effectuées par les utilisateurs sur le réseau. Il cherche à découvrire des bitcoins, c'est-à-dire un nœud fait un très grand nombre de calcul pour résoudre un problème mathématique, il va essayer de découvrir quel est le contenu hashé qui correspond à une suite de nombres qui démarre par un certain nombre de « 0 ». Les transactions vérifiées sont rassemblées dans un bloc qui sera ajouté ensuite à la blockchain. La vérification des transactions requiert de la puissance de calcul [25]. Les mineurs font leurs travaux pour prouver qu'ils travaillent et pour rester allumer. Dans les blockchains publiques devenir mineur est ouvert à tous, en d'autre part dans les blockchains privées le droit de validation des transactions est réservés à certains utilisateurs. Dans Bitcoin les mineurs sont récompensés pour le temps et la puissance de calcul investis (la récompense sera donnée pour celui qui trouve la solution en premier). Actuellement cette récompense est de 12,5 bitcoins par bloc.

2.7 L'écosystème blockchain

Le schéma suivant représente l'écosystème d'une blockchain :

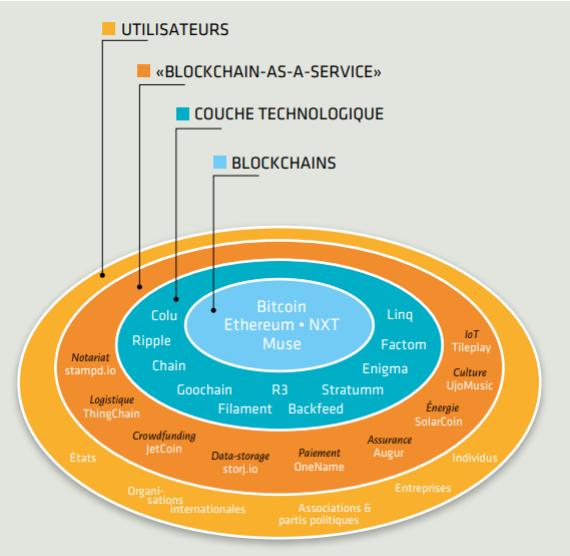


FIGURE 2.3 – L'écosystème blockchain.

2.7.1 Les blockchains

Les différentes blockchains sont les « livres de comptes » qui enregistrent les utilisateurs et les transactions d'un service donnée. Une blockchain peut posséder des spécificités techniques qui favorisent des types d'applications particulières. C'est le cas de la Blockchain « MUSE »pour la rémunération des droits d'auteurs [19].

FIGURE 2.4 – Les blockchains.

2.7.2 La couche technologique

Ces entreprises agissent comme l'interface technique entre une blockchain et les services qu'elles proposent. Elles traitent les informations contenues

dans une blockchain pour les rendre actionnables par des services tiers [19]. Exemple de couche technologique : Colu, Ripple, Linq, etc.

FIGURE 2.5 – Les couches Technologiques.

2.7.3 Blockchain-As-A-Service

Il s'agit d'applications utilisées directement par l'utilisateur final : leur fonctionnement technique est transparent [19]. Exemples de blockchain-as-a-service : JetCoin, OneName, Augur, etc.

FIGURE 2.6 – Exemples d'applications blockchain-as-a-service.

2.7.4 Utilisateurs

Ce sont les structures étatiques, acteurs privés, associations et particuliers qui bénéficient des services blockchain. En d'autres termes, ce sont les participants possédant les autorisations nécessaires pour rejoindre le réseau et effectuer des transactions avec d'autres participants [19].

2.8 Les algorithmes de sécurité utilisés dans les blockchains

Le protocole de Nakamoto repose sur la signature électronique à clé publique fondée sur deux outils cryptographiques : les algorithmes asymétriques et les algorithmes de hashage. Tous les deux sont des fonctions à sens unique, c'est-à-dire qu'ils peuvent aisément être calculés, mais impossibles à inverser [21].

2.8.1 L'utilisation de cryptographie asymétrique

Chaque transaction a recours à la cryptographie asymétrique dont chaque utilisateur possède une paire de clés (publique/privée) pour chiffrer ou signer un message. Dans le cadre du Bitcoin, seule la fonction de signature est utilisée.

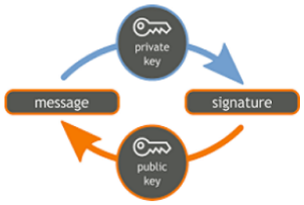


FIGURE 2.7 – Le chiffrement et la signature par la paire de clés.

Dans ce système, un utilisateur crée une suite aléatoire de chiffres « clé privée» à partir duquel une clé publique est générée en utilisant un algorithme de signature numérique à clé publique. Pour le bitcoin, il s'agit d'un algorithme de signature numérique à clé publique dit « à courbe elliptique ECDSA (*Elliptic Curve Digital Signature Algorithm*) ». Par la suite, cette clé privée permettra de signer un message. Les autres utilisateurs du réseau qui connaissent la clé publique correspondante pourront alors vérifier qu'il est bien l'auteur de ce message. La clé publique ne permet pas de retrouver la clé privée qui en est à l'origine ce qui ne risque pas sa diffusion dans le réseau.

Algorithme ECDSA

Soit G un élément d'une courbe elliptique d'ordre n, avec n un nombre premier plus grand que 2^{160}. La courbe est également définie par deux éléments a et b qui sont des éléments d'un corps fini de cardinalité q. Soit le message m à signer.

Génération des clés ECDSA

— Choisir un entier X entre 1 et n-1.
— Calculer Y=X*G en utilisant l'élément G de la courbe elliptique.
— La clé public est Y, et la clé privée est X.

Calcul et vérification d'une signature ECDSA

Après avoir calculé le condensé M du message avec la fonction de hashage SHA1 [2], la signature va se composer de deux parties sig1 et sig2. On choisit d'abord de manière aléatoire un nombre k entre 1 et n-1, puis on calcule sig1 en calculant le point k*G et ne retenant que x (mod n) :
$sig_1 = (k * G)_x$ [mod n]
Si le calcul du modulo donne zéro, il faut recommencer avec une autre valeur de k.
On calcule ensuite sig2 comme suit : $sig_2 = K^{-1}$ (M + X * sig_1) [mod n] où K^{-1} est l'inverse de k mod n.
Le destinataire qui reçoit le message et la signature (sig1, sig2) peut vérifier l'authenticité du message en calculant d'abord son condensé M via SHA1 puis les nombres :
w = sig_2^{-1} [mod n]
u_1 = M * w [mod n]
u_2 = sig_1 * w [mod n]
enfin en calculant le point : (x,y) = u_1 * G + u_2 * Y
La signature est vérifiée si : sig_1 = x [mod n] [26]

2.8.2 La fonction de hachage SHA-256

La fonction de hachage SHA-256 (*Secure hash algorithme 256*) est la fonction de hachage la plus utilisée dans les blockchains. C'est une fonction mathématique qui permet de convertir un ensemble de données numériques en une empreinte (hash). La taille de l'empreinte est fixée quelque soit la taille de la donnée entrée.

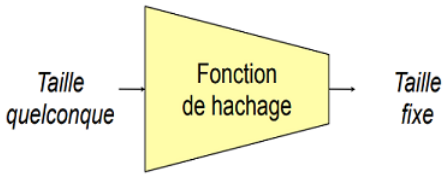


FIGURE 2.8 – Schéma représentant la taille du hash.

La fonction de hachage SHA-256 est une fonction unidirectionnel, c'est-à-dire on peut pas obtenir les données en entrée à partir de leur hash.

2. algorithme de hashage Secure Hash Algorithm

- **Fonctionnement de SHA-256**
  - Compléter le message M à haché pour qu'il soit d'une taille multiple de 512 bits.
  - Découper le message M complété en N blocs de 512 bits. Chaque bloc est ensuite découpé en 16 mots de 32 bits.
  - Calculer le hash de chaque bloc en utilisant des fonctions logiques. Ces fonctions travaille sur les mot de 32 bits du bloc en question.
  - Le hash de 256 bits du message M est obtenu par concaténation des hashs des blocs de M [27].

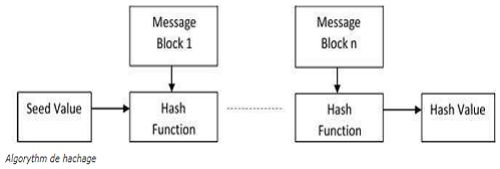


FIGURE 2.9 – Fonctionnement de SHA-256.

2.9 Concepts principaux d'une blockchain

2.9.1 Le portefeuille

Un logiciel client utilisant la cryptographie asymétrique qui permet de se connecter et d'interagir avec la blockchain. L'utilisateur constitue un nœud de la blockchain qui effectue sa synchronisation (chaque nœud possède une copie de la blockchain). Quelques portefeuilles pour l'utilisateur :

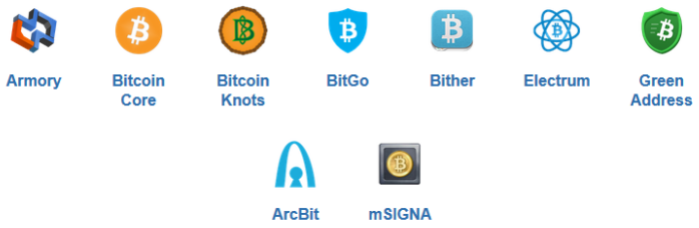


FIGURE 2.10 – Figure représentant quelques portefeuilles pour l'utilisateur.

2.9.2 Le bloc dans la blockchain

Les informations inscrites dans un bloc sont structurées selon des règles bien précises. Le nombre de transactions inscrites et la taille du bloc peuvent varier. Un bloc Bitcoin contient environ 500 transactions et possède une taille de 1 MB. Dans Bitcoin cash, la taille d'un bloc est de 8MB.
Un bloc comporte en générale 3 composantes : l'entête, l'identificateur, et un arbre de Merkel.

- **L'entête du bloc (*bloc header*)**
  L'entête contient les méta-données à propos du bloc concerné, on trouve [28] :

  — **Le hash du bloc précédent :** une valeur de hachage de 256 bits qui pointe vers le bloc précédent. Dans une blockchain chaque bloc est l'héritié du bloc précédent. Cela est du au fait qu'il utilise le hash du bloc précédent pour créer son propre hash.

  — **Horodatage (timestamp) :** heure actuelle en secondes dans le temps universel depuis le 1er janvier 1970.

  — **nBits (hash target) :** seuil cible d'un hachage de bloc valide. Un nombre de 256 bits, qui est utilisé comme seuil pour l'exploration. Afin d'exploiter un bloc avec succès.

  — **Nonce (index) :** un champ de 4 octets (entier non signé de 32 bits), qui commence généralement par 0 et augmente pour chaque calcul de hachage, se produit une seule fois (pour une seule transaction)permettant de trouver un hachage valide pour le bloc afin de garantir que les anciennes communications (transactions) ne peuvent pas être réutilisées.

  — **Hachage de racine de l'arbre de Merkle :** il s'agit d'un arbre structurant les données présentes dans le bloc.

- **L'identificateur du bloc (*block identifier*)**
  Afin d'identifier un bloc, celui-ci possède une signature digitale, appelée hash cryptographique. Dans le cas de Bitcoin, on obtient cette signature digitale en hachant deux fois l'entête du bloc avec l'algorithme SHA256 [28].

- **L'arbre de Merkle (*Merkle tree*)**
  Toutes les transactions sont agencées dans une structure que l'on appelle arbre de Merkle. Dans un tel arbre, les transactions sont prises deux à deux et puis hachées entre elles. L'opération est répétée jusqu'à obtenir un seul hash. Dans le cas du Bitcoin, l'algorithme de hachage utilisé est le SHA-256. Le seul hash obtenu est appelé racine de Merkle, il sera stocké dans l'entête du bloc [28].

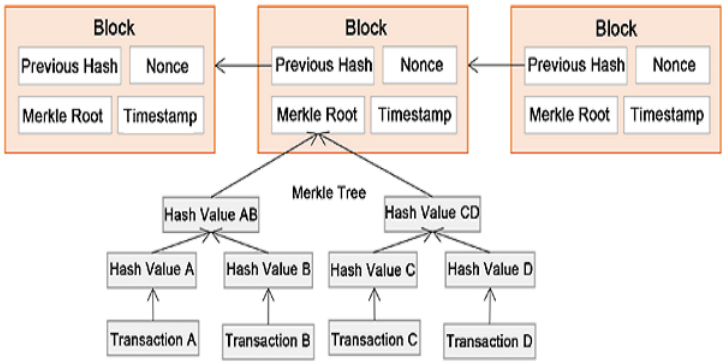


FIGURE 2.11 – Structure d'un bloc.

2.9.3 Les transactions dans la blockchain

Une transaction est un transfert de valeur (monnaie virtuelle, jeton, etc.) entre deux comptes (portefeuilles) inclus dans la blockchain.

La structure d'une transaction

Dans ce qui suit, la structure représentée est celle d'une transaction Bitcoin. En termes simples, une transaction indique au réseau que le propriétaire d'un certain nombre de bitcoins à autorisé leur transfert vers un autre propriétaire. Le nouveau propriétaire peut alors dépenser ces bitcoins en créant une nouvelle transaction qui autorise leurs transfère, et ainsi de suite. La transaction Bitcoin se compose de deux tables [29] :

- Une table d'entrée (*Inputs*) qui répertorie les actifs (un montants en bitcoins) reçu d'un ou plusieurs payeurs.
- Une table de sortie (*Outputs*) qui répertorie les actifs qui seront affectés à un ou plusieurs bénéficiaires.

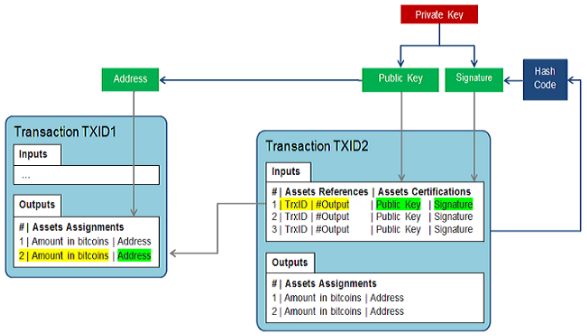


FIGURE 2.12 – Structure d'une transaction.

Lorsqu'un bénéficiaire souhaite utiliser un actif en tant que payeur, une nouvelle transaction est créée. Chaque entrée de la table d'entrée fait référence à un actif d'une entrée de la table de sortie d'une transaction précédente, ce qui assure la légitimité de la transaction. La référence au transactions précédentes permet de savoir la source des bitcoins utilisées dans la transaction courante, et aussi de s'assurer que les bitcoins n'ont pas été dépensés précédemment.

Les formes de transaction

- **La transaction la plus commune**
  Cette transaction est un simple paiement d'une adresse vers une autre. Comme il est difficile d'envoyer le montant exact spécifié des fonds au destinataire, le reste des fonds est renvoyé vers l'expéditeur, via un changement d'adresse du fait qu'un utilisateur ne peut pas faire un paiement vers lui même, il change l'adresse de son portefeuille pour recevoir le reste des fonds [30].

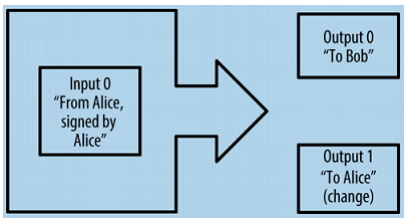


FIGURE 2.13 – Transaction la plus commune.

- **Transaction agrégeant des fonds**
  Ce type de transaction est l'agrégation de plusieurs entrées vers une seule sortie. Les Transactions avec entrées multiples surviennent généralement si le portefeuille de l'utilisateur n'a pas une seule entrée assez grande pour remplir le montant de paiement. Le portefeuille choisit alors d'extraire des bitcoins de plusieurs adresses pour les dépenser[30].

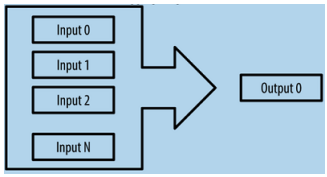


FIGURE 2.14 – Transaction a entrées multiples.

- **Transaction permettant de répartir des fonds**
  Cette transaction distribue une entrée vers de nombreuses sorties qui représentent plusieurs destinataires [30].

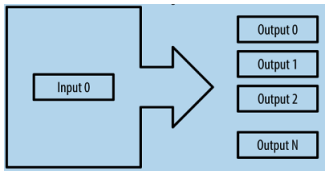


FIGURE 2.15 – Transaction permettant de répartir des fonds.

Construire une transaction

Le portefeuille utilise des algorithmes permettant de sélectionner les entrées et les sorties afin de créer une transaction. L'utilisateur n'a qu'à choisir une destination est un montant, le portefeuille fait ensuite le reste sans lui montrer les détailles du processus.

2.9.4 Contrat intelligent (*Smart Contract*)

Un contrat intelligent est un programme autonome qui, une fois démarré, exécute automatiquement des conditions définies au préalable (un accord ou ensemble de règles), conçus et écrits à l'aide d'un langage de programmation de haut niveau. Il fonctionne comme toute instruction conditionnelle de type « if - then » (« Si » condition vérifiée « Alors » conséquence s'exécute). Ce

Smart Contract est stocké dans la blockchain et mis en œuvre dans le but d'apporter une sécurité supérieur, et de réduire les coûts et les délais de vérification et d'exécution par rapport aux contrats traditionnels. Ces derniers nécessite une validation humaine pour vérifier les termes et conditions, et décider des étapes suivantes, ainsi que l'intervention d'un tiers qualifié de confiance pour appliquer la loi. Pour cela les contrats intelligents sont un outil très puissant pour connecter le monde numérique et le monde physique de manière fiable, sans passer par un contrôle humain couteux et imparfait [20].

2.9.5 Les nœuds

Les nœuds sont des ordinateurs reliés au réseau. Chaque nœud comporte une copie de la base de données qui trace l'historique de l'ensemble des transactions effectuées pour former une chaine de blocs reliés entre eux, ce qui rend la blockchain infalsifiable.

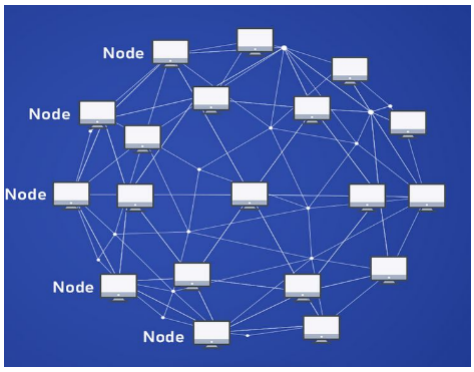


FIGURE 2.16 – les nœuds dans une blockchain.

2.9.6 Le minage (*mining*)

Le Minage (*mining*) est le processus par lequel sont validés et sécurisés les différents blocs. Chaque transaction est chiffrée et stockée dans un bloc, qui peut contenir plusieurs transactions distinctes. Un bloc comporte un marquage numérique (calculs mathématiques) issu du bloc précédent pour vérifier et valider les blocs de transactions. Cette opération de marquage est assurée par des utilisateurs volontaires « Mineurs » qui mettent à disposition leurs temps et la puissance de calcul de leurs ordinateurs pour administrer la blockchain (*Mining pools* ou coopératives de mineurs). Cette opération

permet aux mineurs d'être rémunérés en bitcoin. La valeur du bitcoin est maintenue par des logiciels qui adaptent l'intensité des calculs au nombre de mineurs actifs. Plus il y a de mineurs de Bitcoin, plus les calculs sont complexes et plus la blockchain est sure [23].

2.9.7 Les algorithmes de consensus

Selon Bilal Chouli le fondateur de Neurochain [3], il ya forcément une à vingt secondes de latence pour que le bloc se diffuse dans tout le réseau. Du fait de cette latence plusieurs blocs valides pourraient être crées simultanément par les mineurs. Les nœuds ajoutaient l'un ou l'autre de ces blocs et le réseau comprendrait alors des registres à des états différents [21].

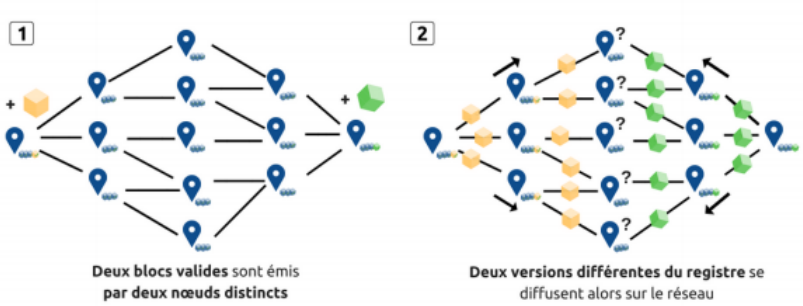


FIGURE 2.17 – Introduction simultanée de deux blocs valides.

Il est donc nécessaire que les nœuds s'accordent sur le prochain bloc à ajouter à la chaîne, c'est pourquoi les protocoles de blockchain prévoient une méthode de consensus.

3. Première infrastructure de Blockchain basée sur l'intelligence artificielle et le machine learning.

La preuve de travail (*proof-of-work "POW"*)

La preuve de travail est l'algorithme qui valide la création de nouveaux blocs. Les mineurs remportent des bitcoins après la validation de chaque bloc. En plus, les mineurs prélèvent des frais sur les transactions qu'ils intègrent à chaque nouveau bloc qu'ils créent. Le montant de ces frais en théorie est déterminé librement par les utilisateurs, mais les mineurs sélectionnent en priorité les plus élevés.

- **Fonctionnement**
  La preuve de travail consiste à demander aux mineurs de résoudre un problème mathématique complexe « puzzle » nécessitant une puissance de calcul informatique importante. Le mineur doit calculer un hash (comporte généralement 64 caractères alpha-numérique), avec la fonction de hachage SHA-256, à partir des données du bloc et d'un aléa « nonce ». Le hash doit être difficile à trouver et facile à vérifiée[31].
  Le hash doit être inférieur à la difficulté de minage, qui est réajustée par le réseau tous les 2016 blocs. Pour y arriver, il faut calculer beaucoup de hash en faisant varier le nonce.
  Le premier mineur qui trouve le hash sera élu pour enregistrer son bloc dans la blockchain. Il diffuse ensuite le nonce et le bloc aux nœuds du réseau. Ces derniers vérifient la validité du bloc en calculant son hash avec le nonce reçu, puis le comparant au hash trouvé par le mineur propriétaire du bloc. Dans le cas d'une validité avérée, les nœuds enregistrent le bloc dans leur registre local.

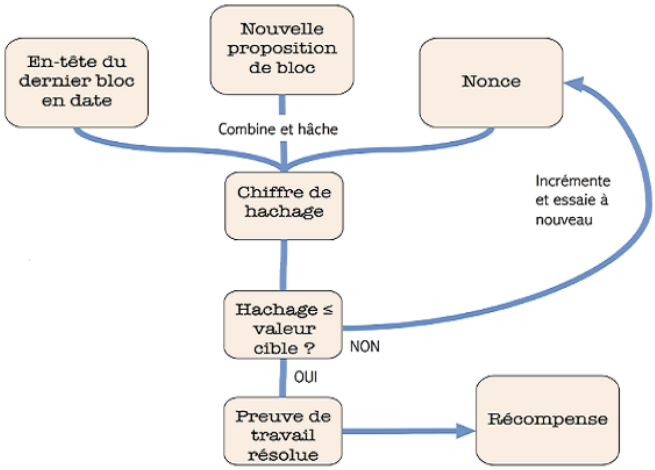


FIGURE 2.18 – Algorithme proof of work.

La preuve d'enjeu (*Proof-Of-Stake "POS"*)

Du fait que la preuve de travail est assez énergivore, pour cela la preuve de participation ou d'enjeu(*proof of stake*) est apparue pour être une alternative à la preuve de travail.

- **Fonctionnement**
  - L'algorithme POS place toutes les transactions non valide dans un pool.
  - Tous les nœuds qui prétendent devenir validateurs pour le bloc suivant dépose une quantité de crypto-monnaie.
  - Un algorithme de sélection combine la quantité de mise (quantité de crypto-monnaie) avec d'autre facteurs (pour rendre la sélection équitable pour tout le monde sur le réseau) et sélectionne un validateur.
  - Le validateur sélectionné vérifie toutes les transactions et publie le bloc, son enjeu reste toujours verrouillé et la récompense n'est pas encore accordée.
  - Si le bloc est validé par les nœuds du réseau, le validateur récupère la mise et la récompense.
  - Si le bloc n'est pas validé par les nœuds du réseau, le validateur perd sa mise et est marqué comme mauvais par l'algorithme [32].

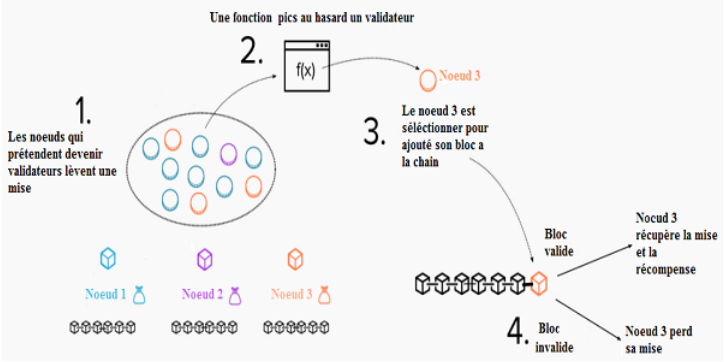


FIGURE 2.19 – Algorithme proof of stake.

Preuve de capacité (*Proof Of Space*)

La preuve de capacité est un protocole pour vérifier et valider une transaction en allouant de l'espace disque inutilisé. Elle est très similaire à la preuve de travail, sauf qu'au lieu de la puissance de calcul, on utilise le stockage (espace disque) de l'ordinateur de l'utilisateur.

Preuve d'autorité (*Proof-Of-Autority "POA"*) :

La preuve d'autorité est une méthode de consensus qui donne à un nombre restreint et désigné de nœuds, qui ont prouvés leurs autorités par une authentification préalable, le pouvoir de valider des transactions et générer de nouveaux blocs directement sans vérification.

- **Fonctionnement**
  - Le nouveau bloc est généré par un nœud leader dans un intervalle de temps.
  - Le rôle de leader est transmis au nœud de validation suivant dans la liste des nœuds de validation.
  - Le nœud leader collecte toutes les nouvelles transactions de sa file d'attente de transactions.
  - Le nœud leader exécute les transactions une par une. Les transactions non valides ou non exécutables sont rejetées.
  - Le nœud leader crée et signe, avec la clé privé du nœud (algorithme ECDSA), un bloc avec les transactions valides.
  - Le nœud leader envoie le bloc créé à d'autre nœuds validateurs.
  - Les autres nœuds validateurs reçoivent le bloc créé et refont le même processus.
  - Si la validation du bloc réussit, alors le nouveau bloc sera ajouté à la chaine de blocs du nœud.
  - Si la validation du bloc a échoué, le nouveau bloc sera rejeté et une transaction de bloc incorrecte sera envoyé.
  - Si le nœud de validation qui a créé ce bloc non valide continue de générer des blocs incorrecte, il peut être exclu de la liste des nœuds validateurs.

2.10 Blockchain : comment ça marche ?

D'un point de vue technique, les membres de la blockchain sont des ressources informatiques (par exemple : des ordinateurs), qui sont préalablement connectés à la blockchain, suite à une phase d'enrôlement. Ces ressources sont couramment appelées des nœuds du fait qu'elles sont mise en réseau au travers internet.

2.10.1 La phase d'enrôlement dans la blockchain

Pour participer aux activités d'une blockchain, une personne doit enrôler un de ses équipements informatiques comme nœud de la blockchain. Au cours de cette opération le nœud (régulier ou mineur) télécharge un logiciel qui lui permet de s'interfacer avec la blockchain. Ce logiciel est personnalisé avec un numéro de compte blockchain (exemple : adresse Bitcoin), et un jeu de clés (publique/privée) qui interviennent dans le processus de signature numérique. Il est impératif de conserver précieusement la clé privée qui joue le rôle de mot de passe pour l'accès au compte [33]. En pratique, il existe un lien évident et facilement vérifiable entre le numéro de compte, la clé publique et la clé privée, comme c'est le cas pour l'adresse Bitcoin qui correspond au résultat du hachage de la clé publique associe, qui correspond à son tour au résultat du hachage de la clé privée.

FIGURE 2.20 – Relation entre clé privée et clé publique et adresse bitcoin.

2.10.2 La phase de transaction

Une blockchain est un ensemble de transactions individuelles regroupées en blocs. Chaque transaction est signée en faisant recours à la cryptographie asymétrique. Elle fonction, pour chaque nœud avec la pair de clés privée/publique. Chaque transaction est signée avec la clé privée du compte émetteur.
Une fois signée, la transaction est placée sur un nœud quelconque du réseau qui la diffuse à son tour, de proche en proche, sur tous les nœuds du réseau, et se met en attente de validation. La signature empêche toute modification de la transaction. La clé public permet à un autre nœud du système de vérifier l'authenticité de la signature [30].
Techniquement, pour générer une signature, le signataire commence par appliquer une fonction de hachage sur les éléments de la transaction, puis de chiffrer le résultat obtenue avec sa clé privée.

2.10.3 La phase de validation d'une transaction

Lorsqu'une nouvelle transaction est émise pour être validée, elle est propagée parmi les participants de la blockchain. Comme la blockchain est fondée sur des échanges pair à pair, n'importe quel nœud qui reçoit une transaction et qu'il n'est pas déjà vu, va immédiatement la propager à tous les nœud auxquels il est connectés. De cette façon, les transactions se propagent rapidement entre les nœuds. Chaque mineur de la blockchain maintient une liste temporaire des transactions non vérifiées dans laquelle il stocke les transactions reçues.
Dans Bitcoin, les transactions non vérifiées sont conservées dans une zone de mémoire chez les mineurs appelée « mempool ». Les mineurs vérifient l'authenticité et la légitimité de chaque transaction, en vérifiant si la signature électronique apposé par l'émetteur de la transaction est valide, et en se référant à l'historique des transactions enregistrées depuis l'origine dans la blockchain. Une fois la transaction est vérifiée, elle est ajouté aux bloc en cours de construction.
Dans Bitcoin, si la taille de mempool des transactions en attente est plus grande que la taille limite d'un bloc, alors les mineurs doivent choisir les transactions à inclure dans le bloc qu'il construisent. Puisque les mineurs ne travaillent pas bénévolement, ils valident les transactions qui maximisent le ratio entre les frais et la taille de la transaction.

2.10.4 La phase d'ajout d'un bloc à la chaine

Les mineurs agglomèrent sous forme d'un bloc les transactions valides et tentent de valider le bloc. Toutes les 10 minutes en moyenne, les mineurs construisent un nouveau bloc qui contient toutes les transactions depuis le bloc précédent.
La validation d'un bloc se fait suivant un algorithme de consensus, qui correspond dans la majorité des cas à la résolution d'un problème mathématique complexe. Ce travail de résolution de problème s'appelle « minage ». Comme plusieurs mineurs tentent de valider le même bloc en même temps, alors le mineurs qui a terminé le minage en premier ajoute le bloc à sa copie du registre et diffuse le bloc ainsi que la solution trouvée aux autre nœuds du réseau.
Quand les autres nœuds reçoivent le bloc et la solution, ils vérifient que ce nouveau bloc est valide, c'est-à-dire qu'ils veillent en particulier à ce que la solution soit valide. Si le bloc est valide , ils l'intègrent alors à leur copie local du registre.
Quand un mineur reçoit un bloc du réseau Bitcoin, cela veut dire qu'il a perdu la course pour ce bloc, il commence alors immédiatement une nouvelle course pour en miner un nouveau.
Dans Bitcoin, chaque mineur ajoute aussi aux nouveaux blocs une transaction spéciale qui paie une récompense vers l'adresse du mineur, s'il trouve une solution qui valide ce bloc en premier, il gagne la récompense car il sera son bloc.

2.10.5 La phase de gestion de bifurcation

Puisque plusieurs blocs étant minés en même temps, que se passe-t-il lorsque deux mineurs finissent en même temps de miner un bloc de transaction ? Il se crée alors ce que l'on nomme une bifurcation (figure 1.20). Les mineurs continuent à miner le long des deux branches, avec potentiellement des nouvelles sous-branches qui apparaissent lorsque deux mineurs terminent le minage de deux blocs en même temps. Cependant, dés qu'un bloc est terminé avant les autres le long d'une branche qui devient la plus longue, tous les mineurs vérifient le contenu du bloc puis doivent mettre à jour leur registre local et bascule sur cette nouvelle branche légitime [34].

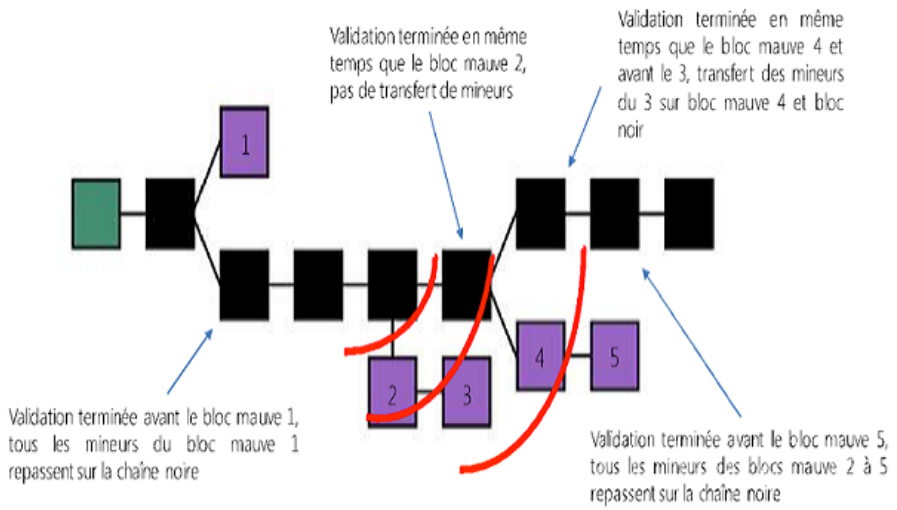


FIGURE 2.21 – Gestion des bifurcations.

2.10.6 Schéma qui résume le fonctionnement d'une blockchain

Dans la blockchain, toutes les transactions sont regroupées sous la forme de blocs. Chaque bloc doit ensuite être validé par les nœuds du réseau en utilisant une méthode algorithmique. Une fois que le bloc est validé, il est ajouté à la chaîne de blocs et devient donc visible de tous les utilisateurs. Voici un schéma qui vous permet de résumé le fonctionnement d'une blockchain.

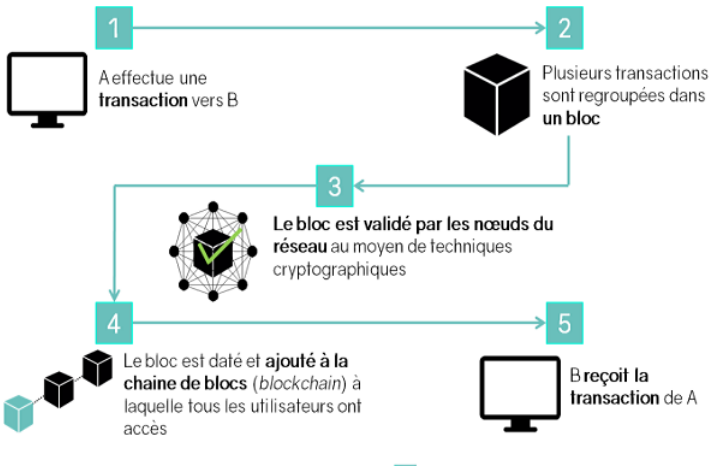


FIGURE 2.22 – Fonctionnement d'une blockchain.

2.11 Domaine d'application et types d'usages de la blockchain

Aujourd'hui le protocole du blockchain a révolutionné pratiquement tout les domaines, dont on trouve :

- **La banque**
  C'est le domaine le plus évident. Certaines banques sont déjà en train d'expérimenter la blockchain notamment pour les applications de back-office où la technologie pourrait se substituer aux employés [20].

- **Les paiements et les transferts d'argent**
  Permet de faire des transactions plus rapides, plus fluides et plus sûres.

- **La cybersécurité**
  La blockchain élimine le « maillon faible » en matière de sécurité informatique : l'homme. Par ailleurs, il est plus difficile pour les cybercriminels de passer inaperçu dans un système ouvert.

- **L'authentification des diplômes**
  L'enregistrement sur un registre ouvert des diplômes, sur le même principe qu'un acte notarié [34].

- **Le vote**
  La blockchain peut être une solution de sécurisation du vote en ligne par exemple [35].

- **La location ou vente de voitures**
  Des expériences existent dans ce domaine où les contrats de vente et d'assurance sont produits en ligne sur le principe « clique, signe et conduis ».

- **L'internet des objets**
  La blockchain devient le centre d'administration et de communication autonome de nos objets connectés : mise à jour logiciel, correction des bugs, communication avec d'autres objets connectés, etc.

- **Les contrats intelligents**
  L'exécution automatisée d'un contrat ou d'une transaction selon des règles définies.

- **La musique en ligne**
  Un streaming plus juste et plus transparent où les musiciens pourraient être payés directement par ceux qui écoutent leur musique.

- **Le co-voiturage**
  L'anti Uber ou Blablacar. Les utilisateurs et prestataires sont en relations directement sans intermédiaires et règlent leurs transactions de façon sécurisée [20].

2.12 Avantages de la blockchain

Les caractéristiques de la blockchain présentent plusieurs avantages :

- **Distribué**
  Etant donné que les données de la blockchain sont souvent stockées dans des milliers de périphériques sur un réseau de nœuds distribué, le système et les données sont extrêmement résistants aux défaillances techniques et aux attaques malveillantes.

- **Stabilité**
  Il est très peu probable que les blocs confirmés soient inversés, ce qui signifie qu'une fois les données enregistrées dans la blockchain, il est extrêmement difficile de les supprimer ou de les modifier.

- **Système sans confiance**
  Un système de blockchain élimine le risque de faire confiance à une seule organisation et réduit ainsi les coûts globaux et les frais de transaction en éliminant les intermédiaires ou les tiers.

2.13 Enjeux et limites de la blockchain

Comme toutes technologies, la blockchain présente aussi des limites d'utilisation, dont on trouve :[20]

- **Les attaques à 51%**
  Une telle attaque peut se produire si une entité parvient à contrôler plus de 50% de la puissance de hachage du réseau, ce qui permettrait éventuellement de perturber le réseau en excluant ou en modifiant intentionnellement l'ordre des transactions [35].

- **Modification de données**
  Modifier les données ou le code d'une chaîne de blocs est généralement très exigeant et nécessite souvent une bifurcation dure, une chaîne étant abandonnée alors qu'une nouvelle est formée.

- **Clés privées**
  Les utilisateurs ont besoin de leur clé privée pour accéder à leurs fonds, ce qui signifie qu'ils agissent comme leur propre banque. Si un utilisateur perd sa clé privée, l'argent est concrètement perdu.

- **Inefficacité**
  Les blockchains, en particulier celles qui utilisent la Preuve de Travail, sont très inefficaces dans un certain sens. Alors que le minage est très compétitif et qu'il n'y a qu'un seul gagnant toutes les dix minutes, le travail de tous les autres mineurs est ainsi gaspillé. Comme les mineurs cherchent continuellement à augmenter leur puissance de calcul, pour avoir plus de chances de trouver un hachage de bloc valide, les ressources utilisées par le réseau Bitcoin ont considérablement augmentées ces dernières années et consomment actuellement plus d'énergie que de nombreux pays, comme le Danemark, l'Irlande ou le Nigéria.

- **Espace de stockage**
  Les registres sur la Blockchain peuvent devenir très volumineux avec le temps. La blockchain Bitcoin nécessite actuellement environ 200 Go de stockage. La croissance actuelle de la taille de la blockchain semble dépasser celle des disques durs et le réseau risque de perdre des nœuds si le registre devient trop volumineux pour être téléchargé et stocké par les utilisateurs.

2.14 Conclusion

La technologie blockchain offre des avantages incroyables par rapport zux bases de données conventionnelles. Elle a permis l'existence de la crypto-monnaie tel que le bitcoin. Pour les consommateur, le bitcoin offre des options de paiement en pair-à-paire moins chère et plus rapides que celles offertes par les services financiers traditionnels.

Dans ce chapitre, nous avons présenté la blockchain, son écosystème, son fonctionnement, et la liaison avec le bitcoin. Nous avons donné aussi des perspectives d'avenir pour cette technologie.

A première vue bitcoin semble tout à fait sécurisé. Mais en réalité, il s'avère que il est possible de désanonymiser les clients ou les transactions dans ce réseau. Dans le chapitre suivant nous présentons les principales techniques de « désanonymisation ». Ainsi que voir comment exploiter l'apprentissage automatique, ou plus tot ces algorithmes pour désanonymiser les transactions bitcoins.

Désanonymisation des clients dans le réseau Bitcoin

3.1 Introduction

L'anonymat et la confidentialité sur internet deviennent plus critique que jamais. Pour cela, de nombreuses solutions sont déployées pour améliorer l'anonymat des utilisateurs lors des transactions en ligne. Les plus célèbres de ces solutions est les crypto-monnaies décentralisées. L'un des premiers exemple est le réseaux bitcoin qui offre aux utilisateurs les possibilités d'effectue des transactions en ligne.

La blockchain a connu un intérêt croissant dans le monde universitaire et dans l'industrie ces dernières années. Cependant, la sécurité et la confidentialité des blockchains continuent d'être au centre du débat. C'est le cas pour le bitcoin, qui peut être considerée comme un épée a double tranchant pour la confidentialité. Bitcoin permet de réaliser des transactions sans avoir a divulguer des informations à des intermédiaires. Cependant, la diffusion des transactions publiquement peut divulguer involontairement des informations confidentielles.

La désanonymisation des clients dans le réseau bitcoin est rendu possible, en analysant les informations des transactions existant dans la blockchain bitcoin. Différents travaux mis en œuvre ont démontré cette possibilité avec différents niveaux de réussite.

Dans ce chapitre, nous abordons ce problème en intégrons l'apprentissage automatique. L'idée de notre travail est de construire un modèle d'apprentissage automatique qui puisse associer une transaction à une identité réelle.

Ce chapitre est organisé comme suit. Nous commençons par expliquer le concept d'anonymat dans bitcoin, et lister les différentes méthodes de désanonymisation des transactions dans le réseaux bitcoin. Puis, nous présentons quelques travaux effectué qui étudient la possibilité de désanonymisation des clients bitcoin. Dans la section qui s'ensuit, nous présentons les différents modèles crées, leur architectures, ainsi que les différents hyper-paramètres associés.

3.2 L'anonymat dans la blockchain bitcoin

Bien que la blockchain bitcoin préserve des propriétés de sécurité de base, atteindre un anonymat raisonnable avec bitcoin peut être assez compliqué et un anonymat parfait peut être impossible.

Dans le Bitcoin, les adresses sont des hachages des clés publiques d'un nœud (utilisateur) dans le réseau. Ainsi, l'adresse d'un utilisateur peut être comme une pseudo-identité (pseudonyme), pour protéger le vrai nom de l'utilisateur ou son adresse IP.

Le système Bitcoin fonctionne sur le réseau décentralisé TOR (*The Onion Routing*) qui garantie le secret et l'anonymat des transactions à l'aide des couches de chiffrement, en faisant passer le message par trois serveurs sélectionnés au hasard, où chaque couche contient uniquement la machine qui a envoyé le message et celle qui va le recevoir. Cet anonymat est basé sur les faits suivants [36] :

- L'adresse Bitcoin ne peut pas être mise en correspondance avec l'identité réelle.
- Les transactions Bitcoin ne contiennent aucune information personnelle.
- Les nouvelles transactions sont réparties radialement, pour que l'adresse IP de l'expéditeur ne soit pas exposée.

Cependant, la faiblesse de l'anonymat de Bitcoin se reflète dans les domaines suivants :

- Certains mécanismes d'authentification en nom réel aide les fournisseurs de services Bitcoin à trouver les adresses qui ont été déposées et retirées (lorsqu'un utilisateur achète pour la première fois du bitcoin chez un fournisseur, cas de coinbase par exemple, il exige de vérifier l'identité du client avant d'effectuer la procédure d'achat).
- La chaîne des transactions est transparente et traçable.
- La possibilité de rassembler certaines ou toutes les entrées lors de l'envoi des bitcoins, ce qui peut exposer d'autres adresses de l'expéditeur.

- L'adresse de changement [1] des transactions pourrait être classée par les attaquants à l'expéditeur (les attaquant peuvent déterminer l'adresse principale du client).

Donc la connaissance d'une clé publique d'un utilisateur pourraient, non seulement trouver une transaction effectuer par le propitaire de cette clé, mais de révéler aussi tout l'historique des transactions de l'utilisateur. Si le lien entre la clé publique et l'identité réelle de l'utilisateur est trouvé, alors les transactions seront désanonymisées.

3.3 Désanonymisation dans le réseau Bitcoin

La désanonymisation consiste à lier les adresses de portefeuille au propriétaire qui se cache derrière. En d'autre termes, c'est découvrir la relation entre l'adresse Bitcoin et l'utilisateur, en se basant sur deux méthodes :

1. L'analyse de la chaine de transaction (ATC), en obtenant des informations sur les transactions à partir des données publiques de la blockchain, classer les adresses Bitcoins en fonction de la faiblesse de l'anonymat Bitcoin, et relier les adresses Bitcoins aux identités personnelles. C'est–à-dire étudier la connectivité et le trafic des pairs. Cette méthode est capable de regrouper de nombreuses adresses Bitcoin en un seul utilisateur, ou de trouver des indices pour les connecter aux adresses utilisateurs réelles.
2. L'analyse du protocole et du réseau Bitcoin (ABPN), qui utilise les caractéristiques de propagation des transactions pour déduire l'adresse IP source d'une nouvelle transaction. En d'autres termes, lier des pseudonymes de l'utilisateur et les transactions dans la blockchain réalisées via une analyse graphique et des flux de transaction.

Dans l'ABPN actif ou passif, il existe plusieurs attaques dans lesquelles la nouvelle transaction peut être liée à une adresse IP pour trouver sa relation avec l'adresse Bitcoin [37].

1. lors de l'envoie des bitcoin, la transaction vide le solde de l'adresse expéditeur. Si le montant à envoyer est infèrieur au solde de l'adresse, la transaction doit fournir au réseau une nouvelle adresse pour envoyer la « monnaie »restante

1. **Renifleur de protocole Bitcoin**
   Dans le protocole Bitcoin, les données ne sont pas cryptées, ce qui fait qu'un renifleur bien formé pourrait surveiller toutes les commandes « invitation » sortantes pour vérifier si l'ID de transaction a été déjà manipulé. Si c'est le cas, donc c'est une nouvelle transaction. Par conséquent, on pourra obtenir la relation entre l'adresse Bitcoin et l'adresse IP. Si un tel système de surveillance est déployé à l'entrée du réseau d'une ville (ou avec la complicité du fournisseur d'accés internet), la véritable identité peut être retrouvée à l'aide des enregistrements IP de l'opérateur télécom.

2. **Attaque Sybil**
   L'attaquant pourra se connecter à tous les nœuds du réseau Bitcoin, en sachant que la première adresse IP source d'une nouvelle transaction appartient à l'expéditeur d'origine. Cette méthode engendre à son tour trois problèmes :[37]
   (a) La difficulté d'avoir la véritable adresse IP en utilisant TOR [2].
   (b) Un grand nombre de clients ne peuvent pas être connectés directement, donc l'expéditeur ne peut pas être localisé correctement.
   (c) Le même client possède différentes sessions, différentes adresses IP et différents réseaux.

   Ce qui rend la liaison des véritables adresse IP et les transactions difficile.

3. **Attaque Sybil plus nœuds d'entrée**
   Une méthode qui oblige les nœuds Bitcoin à refuser les connexions des nœuds de sortie de TOR et de détecter le nœud d'entrée d'un client spécifié. Cette méthode, grâce au délai de livraison de toutes les adresses IP mentionnées dans la transaction, permet de trouver l'adresse source d'une nouvelle transaction[37].

4. **Faux nœuds attaquent**
   Une fausse attaque de nœuds Bitcoin (en introduisant les cookies d'adresse), qui fonctionne en établissant un nombre suffisant de faux nœud de sortie TOR (3% de tous les nœuds de sortie du réseau TOR)

2. TOR cache la véritable adresse IP. Une nouvelle étude a découvert une version du navigateur Tor capable de voler les bitcoins des acheteurs sur le Darknet (un type du Dark Web. Un sous-réseau d'internet qui utilise des protocoles spécifiques qui intègrent des fonctions d'anonymisation)

et de faux nœuds Bitcoin (1000 à 1500). Ces nœuds se comportent comme des nœuds normaux en exécutant le code des attaquants. Le cookie d'adresse à son tour identifie le client même s'il utilise différentes adresses IP, différentes sessions et réseaux.[37]

3.4 Les techniques de désanonymisation dans le réseau Bitcoin

La liaison des adresses avec leurs propriétaires du monde réel nécessite beaucoup d'informations externes supplémentaires en plus des données des transactions et leurs historiques. Pour ce faire, on peut utiliser plusieurs techniques divisées en deux catégories : passives et actives.

- **Les méthodes passives**
  Ces méthodes s'appuie sur l'analyse des données pour inférer les informations privées (personnelles) des utilisateurs, basées sur l'observation des nœuds dans le réseau Bitcoin (ATC).

  1. **Correspondance directe**
     Il s'agit de la méthode de désanonymisation la plus simple. Cette méthode consiste à tenter de trouver le propriétaire de l'adresse Bitcoin en la recherchant dans des sources publiques. Ce qui mène à l'obtention de l'identité numérique correspondante en cas de succès. [38].

  2. **Empreintes digitales**
     Le web tracking (technique qui consiste à identifier les visiteurs d'un site web à partir de leurs adresses IP, puis à reconstituer leurs parcours), peut désanonymiser les utilisateurs Bitcoin. Par exemple, un attaquant peut lier un utilisateur web à une transaction sur la blockchain Bitcoin. Un autre exemple est celui des sites d'achats en ligne, si un attaquent connait le prix et l'heure approximatif de l'achat, il peut analyser simplement la blockchain Bitcoin pour trouver la transaction de la victime.

3. **Heuristique à entrées multiples**
   Est une heuristique ou une hypothèse selon laquelle toutes les entrées d'une transaction appartiennent à la même entité.
   Les transactions avec plusieurs entrées (transaction à entrées multiples), surviennent généralement lorsque l'utilisateur souhaite effectuer un paiement et que le montant du paiement dépasse la valeur de chacune des entrées disponibles dans son portefeuille. Les portefeuilles choisiront alors de dépenser plus d'une entrée dans leurs transactions. La conclusion simple est que si ces entrées (les bitcoins à dépensées) appartiennent à des adresses différentes, ces adresses alors appartiennent au même utilisateur [38].

4. **L'heuristique ombre (changement d'adresse heuristique)**
   Le protocole Bitcoin oblige chaque transaction à dépenser, en sortie, la totalité de l'entrée, et comme il est difficile d'envoyer le montant exacte spécifié des fonds au destinataire, le reste des fonds doit être retourné à l'entité émettrice via un changement d'adresse [38].

   — **Réutilisation des adresses**
   Cette méthode détecte la modification de l'adresse en vérifiant les adresses de sortie, s'il existe une adresse qui n'a jamais été observée auparavant, il ya de fortes chances que ca soit une adresse de changement.

   — **Détection utilisant des décimales**
   La longueur des décimales dans la valeur d'une adresse de modification est supposée être beaucoup plus longue que toute autre adresse de sortie.

5. **Clustering des adresses Bitcoins**
   Cette méthode consiste à déterminer les adresses Bitcoin qui appartiennent au même utilisateur en utilisant les deux heuristiques décrites précédemment et une puissance de calcul accrue qui entrainera un traitement plus rapide des transactions [39].
   Le clustering ne permet pas de rattacher des transactions à une personne physique, car les données de la blockchain sont pseudonymes, il vise juste à construire des clusters d'adresses présumés, appartenir à la même personne ou entité.

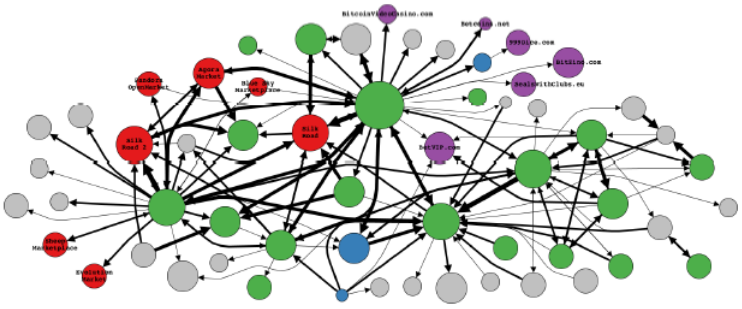


FIGURE 3.1 – Schéma représentant les plus grands clusters d'adresses Bitcoin.

Sur le schéma :
- les sommets rouges sont des marchés *darknet*.
- les sommets violets sont des services de jeu.
- les sommets verts sont des échanges.
- les sommets bleus sont des pools de minage.

Les sommets gris ne sont pas immédiatement identifiables à l'aide d'informations accessibles au public.

- **Les méthodes actives**
  Ces méthodes vise le réseau Bitcoin en analysant les graphiques et les flux des transactions (ABPN).

  1. **Ingénierie sociale (*Social engineering*)**
     Les utilisateurs peuvent être trompés et se diriger vers un site internet de *phishing* ou ils ouvrent leurs portefeuilles chiffrés, et l'attaquant dans ce cas peut facilement liée l'adresse IP à l'adresse du portefeuille.

  2. **Analyse de trafic**
     Un attaquant qui possède des ressources similaires à celle du FAI. (Fournisseur d'accès à Internet), peut espionner la connexion internet d'une victime, et cela révélerait que la victime est un utilisateur bitcoin. Si l'attaquant voit que la victime à initier la diffusion d'une transaction, il sera en mesure de lier l'adresse IP de la victime aux informations bitcoin découvertes.

3. ***Mystery shopper payment***
   Dans le paiement client mystère (*Mystery shopper payment*), l'attaquant envoi une petite somme des bitcoins sur une adresse qu'il sait appartenir à sa cible, ce qui lui donne un point de départ ferme pour commencer l'analyse.

4. ***Forced adress reuse***
   Dans la réutilisation forcée d'adresse (*Forced adress reuse*), l'attaquant envoie une petite quantité de bitcoin à une adresse déjà existante, dans l'espoir que la cible dépensera cet quantité comme output en même temps qu'un autre, ce qui lui permettra d'associer d'autre adresses.

5. **Regroupement des adresses IP**
   Les nœuds d'entrée d'une transaction écoutent les adresses IP client annoncées sur le réseau et enregistrent l'ensemble des nœuds serveurs reliés à cette adresse. Cet ensemble sera composé alors des nœuds d'entrées de ce client et de fausses entrées, qui seront filtrées ensuite en exploitant le protocole Bitcoin, basé sur le fait que le client ne transmet pas d'adresse sur une connexion récemment utilisée pour le transfert de la même adresse. Une fois l'ensemble des nœuds d'entrée déterminé (connu), la méthode propose d'établir plusieurs connexions avec une grande quantité des nœuds de serveur du réseau en écoutant les transactions diffusées. Si une transaction est d'abord diffusée à partir des nœuds qui forment l'ensemble filtré, le créateur de la transaction alors appartient à ces nœuds d'entrée [40].
   Cette méthode intervient uniquement si les nœuds d'entrées restent largement constants pour un client donné. Elle nécessite aussi des ressources importantes pour être exécutée, du fait qu'une grande quantité de connexions doit être établie avec presque tous les nœuds serveurs du réseau.

6. **Cookie d'adresses**
   Cette technique sert à identifier les utilisateurs de Bitcoin en définissant un « cookie d'adresse » sur l'ordinateur de l'utilisateur. Le cookie peut être défini et vérifié même lorsque l'utilisateur se connecte au réseau Bitcoin via Tor. Il peut être utilisé pour corréler différentes transactions du même utilisateur même entre différentes sessions (c'est-à-dire après le redémarrage de son ordina-

teur). Suivant cette technique, si un utilisateur décide de faire une transaction non sensible, c'est-à-dire sans TOR (utiliser un autre réseau), son empreinte digitale peut être corrélée à son adresse IP, désanonymisant (identifiant) ainsi toutes ses transactions envoyées précédemment via Tor.

7. **Déconnexion de TOR**
   Pour établir une connexion à un service ou à un nœud via TOR, l'utilisateur choisit une chaine de trois relais TOR pour acheminer le message vers le nœud cible. Le nœud final de la chaine (TOR exit) apparait au service comme étant à l'origine de cette connexion. A chaque fois qu'un pair reçoit un message mal formé, il augmente le score de pénalité de l'adresse IP d'où provient le message, et interdit cette adresse pendant 24 heures lorsque ce score atteint 100. Pour empêcher les utilisateurs de Bitcoin d'utiliser TOR pour effectuer des transactions, on pourrait exploiter la protection DoS intégrée de Bitcoin, en essayant de se connecter à travers un nœud pour usurper l'adresse IP du nœud de sortie et émettre des messages mal formés à partir de cette nouvelle adresse, ce qui entrainerait l'interdiction de 24 heures du nœud de sortie [41].

3.5 Phases de désanonymisation

Après avoir décrit les différentes techniques utilisées dans la désanonymisation des transactions Bitcoin, passant maintenant à la phase de désanonymisation qui comporte les étapes suivantes [42] :

1. Obtenir la liste « S » de serveurs [3]. Cette liste est régulièrement rafraîchie.
2. Composer une liste « C » de clients Bitcoin pour la désanonymisation.
3. Apprentissage des nœuds d'entrée des clients « C » quand ils se connectent au réseau.
4. Écouter des serveurs depuis « S » et cartographier les transactions aux nœuds d'entrée, puis aux clients.

3. Nœuds qui jouent le role des serveurs dans le réseau Bitcoin

- **Étape 1 : Obtenir la liste des serveurs**
  Cette étape consiste à obtenir la liste complète des pairs en interrogeant tous les homologues connus avec un message « GETADDR » [4]. Chaque adresse P dans le message de réponse « ADDR » [5] peut être vérifier s'il est en ligne par l'établissement d'une connexion TCP et l'envoie d'un message VERSION. Si c'est le cas, P est désigné comme serveur. Cette étape peut être lancé en interrogeant un petit ensemble de nœuds de départ, continuant ensuite avec les adresses IP nouvellement reçues.

- **Étape 2 : Composition de la liste de désanonymisation**
  Cette étape consiste à sélectionner un ensemble C de nœuds qui seront soit dans des plages IP des principaux fournisseurs de services Internet, ou bien collecter des adresses déjà annoncées, ou encore prendre quelques entrées à partir de la liste des pairs obtenue à l'étape 1.

- **Étape 3 : Mappage des clients à leurs nœuds d'entrée**
  Cette étape consiste à identifier les nœuds d'entrée des clients qui se connectent aux réseaux, en utilisant la liste C des adresses et les différentes techniques de désanonymisation vu précédemment (trois nœuds d'entrées identifient de manières unique un utilisateur, bien que deux nœuds le font également pour un grand pourcentage d'utilisateurs).

- **Étape 4 : Mappage des transactions aux nœuds d'entrée**
  Cette étape s'exécute en parallèle avec les étapes 1 à 3. Elle s'appuie sur la corrélation des transactions apparaissant dans le réseau avec les ensembles obtenu à l'étape 2. Elle consiste à écouter des messages « INVENTORY » avec les hachages de transaction reçus et recueillir RT pour chaque transaction T (la première adresse q du serveur Bitcoin qui a transmis le message). Elle compare ensuite l'ensemble corrélé avec les RT et les entrées correspondantes aux paires (P,T), en composant tous les « 3-tuples » possibles pour chercher leur apparitions dans RT. Si on trouve une correspondance alors on obtient ainsi une paire (R, T), sinon on refait le travail pour « 2-tuples », puis « 1-tuples ».

4. GETADDR est une commande utilisée pour demander les adresses des nœuds de la transaction

5. Un message ADDR peut contenir n'importe quel nombre d'adresses, mais les messages contenant plus de 1 000 adresses sont rejetés du côté distant

3.6 Autres travaux connexes

Dans le protocole Bitcoin actuel, l'historique complet des transactions est accessible au public. Tout le monde peut voir comment les bitcoins voyagent d'un pseudonyme à un autre. Donc les différents pseudonymes du même utilisateur peuvent être liés.

Une possibilité théorique d'une telle attaque a déjà été mentionnée dans l'article Bitcoin original [43] en utilisant l'heuristique à entrée multiples.

Depuis, plusieurs articles [44, 45] ont montré qu'il est effectivement possible de regrouper les différents pseudonymes d'un utilisateur en un cluster, en analysant le graphe des transactions. Combiné avec d'autres sources (par ex. les forum sur internet), les clusters peuvent parfois être mis en correspondance avec des identités réelles [46, 44]. Même ainsi, ces méthodes ne sont pas génériques et le problème de la façon de lier une adresse Bitcoin à une identité réelle n'est toujours pas résolu.

L'étude de l'intégralité du trafic IP des paires bitcoin révélerait les origines de chaque transaction et révélerait l'identité de nombreux utilisateurs. Koshy et al. [47], ont été les premiers à tenter une attaque dans ce sens, et on réussi à désanonymiser 1162 adresses sur une période de 5 mois.

Reid et Harrigan [46], ont examinés l'impact des clusters d'adresses construits en utilisant l'heuristique multi-entrée sur l'anonymat.

Moser et al. [48, 49], ont examinés les implications des analyses de la blockchain, y compris le clustering des adresses, pour lutter contre le blanchiment d'argent.

3.7 Motivation et objectifs

Dans le protocole Bitcon, comme déja cité, l'identification de ces utilisateurs a connu toujours des difficultés reliées au degré d'anonymat et la forte puissance des algorithmes utilisés pour l'assurer.
Auparavant, certain heuristiques basées sur des règles structurelles ou des observations des transitions ont été mises en œuvre pour le clustering des adresses Bitcoin. Aujourd'hui d'autres méthodes plus efficaces apparaissent en faisant intervenir l'apprentissage en profondeur.

L'objectif de notre travail est de présenter alors une autre méthode de désanonymisation suivant cette topologie, en exploitant le système des réseaux de neurones artificiels, qui sont presque devenus synonyme d'intelligence artificielle grâce à leur capacité de classification et de généralisation, pour réaliser la cartographie adresse-utilisateur.

Nous commençons notre travail d'abord par la génération d'un modèle de classification des transactions comme étant « licite » ou « illicite », en utilisant les caractéristiques de ces transactions.

Dans notre cas, différents modèles de réseaux de neurones artificiels sont créés en utilisant différents hyper-paramètres et différentes architectures. Ces modèles seront ensuite abordés dans la désanonymisation des transactions.

3.8 Notre approche

Notre travail consiste à désanonymiser les transactions Bitcoin en s'appuyant sur un modèle d'apprentissage automatique. Les modèles crées sont des modèles d'apprentissage supervisé par réseaux de neurones non-bouclés et multi-couches.
Pour expérimenter les différents hyper-paramètres existants, trois modèles d'apprentissage sont proposés. Dans ce qui suit, la description du jeu de données utilisé et les modèles proposés.

3.8.1 Architectures des modèles proposés

Les modèles d'apprentissage automatique crées représentent des modèles de classification. Ils prennent en entrée les caractéristiques des transactions bitcoin et produisent en sortie 2 classes (licite, illicite).

Architecture du modèle 1

Ce modèle est composé d'une couche d'entrée, 1 couche cachée, 1 couche *dropout* et 1 couche de sortie.

La couche d'entrée utilise 167 neurones, le premier neurone reçoit l'id de la transaction. Les 166 neurones restant reçoivent les caractéristiques de la transaction.

La couche cachée, utilises 85 neurones, avec la fonction d'activation *Selu*. Les poids associés à cette couche sont initialisés de manière aléatoire à des nombres proche de zéro.

La couche *dropout* permet de lutter contre le sur-ajustement. On indique que 10% des neurones seront désactivés à chaque itération pendant le processus d'entrainement.

La couche de sortie utilise 2 neurones avec la fonction d'activation *softmax* qui permet de calculer la distribution de probabilité des 2 classes. Les poids de cette couche sont initialisés de manière aléatoire à des nombres proche de zéro.

Le modèle utilise un optimiseur *rmsprop*[6], et utilise l'entropie croisée[7] pour calculé la perte entre les étiquettes réels et les prédictions.

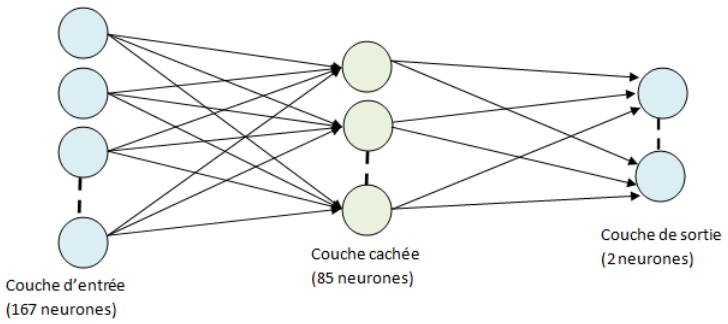


FIGURE 3.2 – Schéma représentant l'architecture du modèle 1.

6. *root mean square propagation*
7. *sparse-categorical-crossentropy*

Architecture du modèle 2

Ce modèle est composé d'une couche d'entrée, 3 couches cachées, 1 couche *dropout* et 1 couche de sortie.

La couche d'entrée utilise 167 neurones, le premier neurone reçoit l'id de la transaction. Les 166 neurones restant reçoivent les caractéristiques de la transaction.

Les 3 couches cachées, utilises 100, 50 et 25 neurones respectivement, avec la fonction d'activation *Tanh*. Les poids associés à ces couches sont initialisés de manière aléatoire à des nombres proche de zéro.

La couche *dropout* permet de lutter contre le sur-ajustement. On indique que 20% des neurones seront désactivés à chaque itération pendant le processus d'entrainement.

La couche de sortie utilise 2 neurones avec la fonction d'activation *softmax*. les poids de cette couche sont initialisés de manière aléatoire à des nombres proche de zéro.

Le modèle utilise un optimiseur *sgd*[8], et utilise l'entropie croisée[9] pour calculé la perte entre les étiquettes réels et les prédictions.

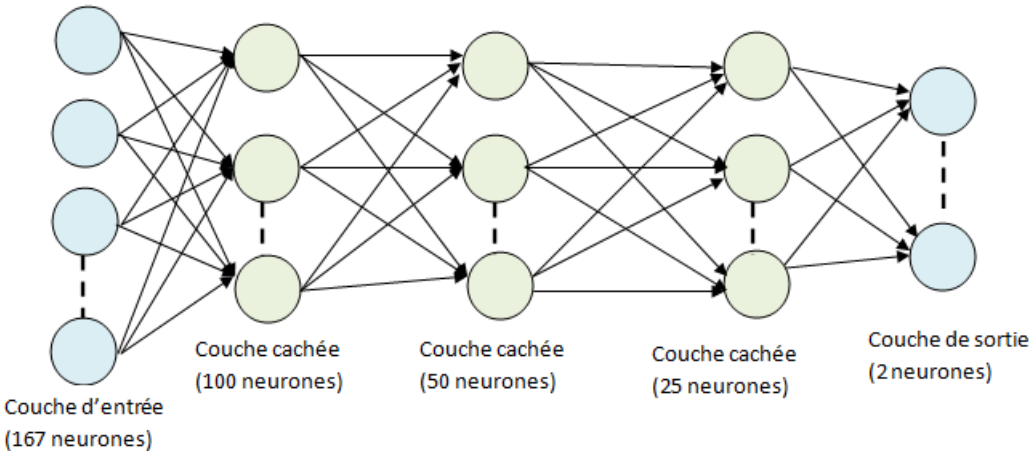


FIGURE 3.3 – Schéma représentant l'architecture du modèle 2.

8. *stochastic gradient descent optimizer*
9. *sparse-categorical-crossentropy*

Architecture du modèle 3

Le troisième modèle est composé d'une couche d'entrée, 5 couches cachées et 1 couche de sortie.

La couche d'entrée utilise 167 neurones, le premier neurones reçoit l'id de la transaction. Les 166 restant reçoivent les caractéristiques de la transaction.

La première couche cachée utilise 100 neurones. La deuxième utilise 70 neurones. La troisième utilise 50 neurones. La quatrième et la cinquième utilises 25 neurones et 11 neurones respectivement. Ces 5 couches cachées utilisent la fonction d'activation *relu*. Les poids associés à ces couches sont initialisés aléatoirement à des nombres proches de zéro.

La couche de sortie utilise 2 neurones avec la fonction d'activation *softmax*.

Pendant le processus d'apprentissage 10% de neurones seront désactivés à chaque itération pour éviter le sur-ajustement, et cela grâce à la couche *dropout*.

Le modèle utilise l'optimiseur *adam*[10],et utilise l'entropie croisée[11] pour calculé la perte entre les étiquettes réels et les prédictions.

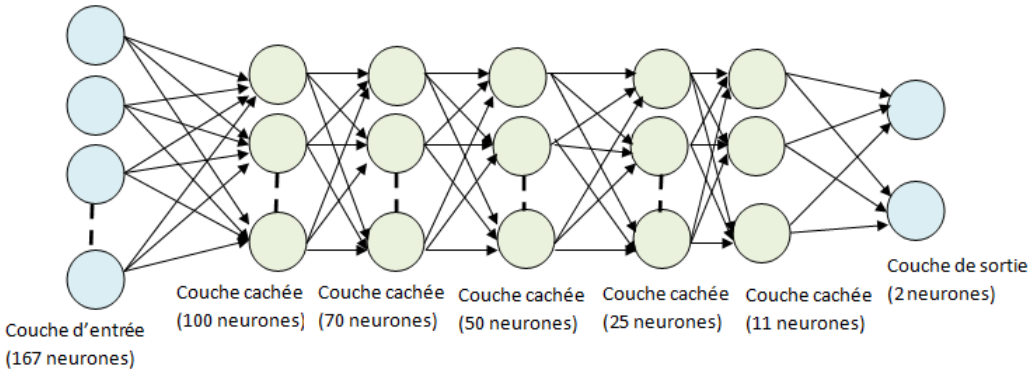


FIGURE 3.4 – Schéma représentant l'architecture du modèle 3

10. une méthode de *stochastic gradient descent optimizer*
11. *sparse-categorical-crossentropy*

3.9 Conclusion

Dans ce chapitre, nous avons mis l'accent sur le problème d'anonymat dans la blockchain bitcoin. Ensuite, nous avons présenté les différentes méthodes de désanonymisation dans le système bitcoin. Enfin, nous terminons par la représentation de notre approche et nous décrivons les architectures des modèles que nous avons réalisés.

Le prochain chapitre décrit notre implémentation et les résultats obtenus par les modèles précédents.

Implémentation

4.1 Introduction

Après avoir vu les différents modèles d'apprentissage automatique proposés pour désanonymiser les transactions effectuées dans le réseau bitcoin dans le chapitre précédent. Nous détaillons dans ce chapitre la phase d'implémentation de ces modèles.

Dans la section qui s'ensuit, nous présentons l'environnement de développement, notamment les logiciels et les librairies que nous avons utilisés. Puis, nous expliquons comment le jeu de données a été divisé lors de la réalisation de nos modèles. A la fin, et avant de conclure, nous montrons les résultats obtenus par chaque modèle proposé et effectuons une comparaison entre eux afin de définir le mieux adapté pour la désanonymisation des transactions bitcoin.

4.2 Langage et environnement de développement

4.2.1 Langage de développement Python

Python [1] est un langage de programmation de haut niveau, orienté objet, à usage général, avec une sémantique dynamique intégrée principalement pour le développement Web et d'application.
Python est un langage simple, facile à apprendre, qui prend en charge l'utilisation de modules et de packages. De ce fait, les programmes peuvent être conçus dans un style modulaire dont le code peut être réutilisé dans une variété de projets. Une fois qu'un module ou package développé, il peut être mis à l'échelle pour une utilisation dans d'autres projets, en exploitant la facilité d'importation et d'exportation de ces modules.

FIGURE 4.1 – Logo de Python

1. https ://www.python.org/

4.2.2 Logiciels

Pycharm

Pycharm [2] est un environnement de développement intégré IDE (*integrated development environment*). Pycharm offre un ensemble complet d'outils pour un développement productif avec le langage de programmation Python. Il offre un éditeur de code intelligent pour écrire des scripts et une console pour exécuter des programmes. Il permet l'analyse de code et contient un débogueur graphique. Il permet également la navigation rapide d'un fichier à un autre.

Pycharm est développé par jetBraines qui est une entreprise informatique éditent des logiciels pour développeur de logiciels.

FIGURE 4.2 – Logo de Pycharm.

4.2.3 Bibliothèques d'apprentissage automatique

Anaconda

Anaconda [3] est une plateforme de science des données, conçu dans le but de permettre aux personnes intéressées par ces domaines d'installer facilement tous (ou la plupart) des packages nécessaires avec une seule installation.

Anaconda est une distribution *open source* pour Python et R, utilisée pour la science des données, l'apprentissage automatique, et l'apprentissage profond avec plus de 300 bibliothèques. Ces bibliothèques sont utilisées pour collecter facilement des données à partir de diverses sources, à l'aide de divers algorithmes d'apprentissage automatique et l'IA. Il aide à obtenir une

2. https ://www.jetbrains.com/fr-fr/pycharm/
3. https ://www.anaconda.com/

configuration d'environnement facilement gérable qui peut déployer n'importe quel projet en cliquant sur un seul bouton.

FIGURE 4.3 – Logo de Anaconda

Tensorflow

Tensorflow [4] est une plateforme *open source*, développé par l'équipe « *Google Brain* » au sein de l'organisation de recherche « Machine Intelligente » de Google, dédiée à l'apprentissage automatique et la recherche sur les réseaux de neurones profonds. Elle est conçu dans le but d'effectuer le calcul numérique à l'aide de graphiques de flux de données comportant des nœuds et des arêtes.
Son nom s'inspire notamment du fait que les opérations actuelles des réseaux de neurones s'effectuent principalement via une table multidimensionnelle de données, appelée Tenseurs (*Tensor*) qui est l'équivalent d'une matrice.
TensorFlow est multiplateforme, qui fonctionne sur tout les GPU et les processeurs y compris les plateformes mobiles et embarquées, et même les unités de traitement tensoriel (TPU), qui sont du matériel spécialisé sur lequel effectuer des calculs tensoriels.

FIGURE 4.4 – Logo de Tensorflow.

4. https ://www.tensorflow.org/

Keras

Keras [5] est une interface de programmation d'application API (*Application Programming Interface*), d'apprentissage en profondeur (une API de réseau de neurones de haut niveau) écrite en Python et interfaçable avec *TensorFlow*, *CNTK* et *Theano*. Elle a été développée dans le but de permettre des expérimentations rapides et être capable d'aller de l'idée au résultat avec le plus fiable délai possible.
Keras est composé alors d'une bibliothèque de composants d'apprentissage automatique couramment utilisés, y compris des objectifs des fonctions d'activation et des optimiseurs. De plus, Keras propose le développement de plateformes mobiles pour les projets d'apprentissage en profondeur sur les smartphone (*IOS*/ *Android*).

FIGURE 4.5 – Logo de Keras.

Scikit-learn

Scikit-learn [6] est une bibliothèque libre pour Python destiné à l'apprentissage automatique. Elle comporte divers algorithme de classification, de régression, de regroupement, etc. Nous avons utilisés le package Sklearn-preprocessing de Scikit-learn pour la mise à l'échelle des données pour améliorer les performances des modèles.

FIGURE 4.6 – Logo de Scikit-learn.

5. https ://keras.io/
6. https ://scikit-learn.org/

4.2.4 Bibliothèques de science des données

Pandas

Pandas [7] est une bibliothèque *open Source* fournissant des structures des données hautes performances et faciles à utiliser, et des outils d'analyse de données pour le langage de programmation Python. Elle propose en particulier des opérations de manipulation de tableaux numériques et des séries temporelles.

FIGURE 4.7 – Logo de Pandas.

Numpy

Numpy [8] est une librairie *open Source* visant à permettre le calcul numérique avec Python. Elle à été crée en 2005 et est destinée à manipuler des matrices ou tableaux multidimensionnels, ainsi que des fonctions mathématiques opérant sur ces tableaux.

FIGURE 4.8 – Logo de NumPy

4.2.5 Bibliothèques de visualisation

Matplotlib

Matplotlib [9] est une bibliothèque du langage de programmation Python destinée à tracer des données sous forme de graphiques. Elle peut être combinée avec la bibliothèque Python de calcul scientifique Numpy.

7. https ://pandas.pydata.org/
8. https ://numpy.org/
9. https ://matplotlib.org/

4.3 Jeu de données utilisé

Nous avons utilisé une collection de données étiquetées de 200 000 transactions bitcoin publiées par la société Elliptic début aout 2019. Elliptic est une société de renseignement sur les crypto-monnaies contre les activités criminelles.
La collection de données contient 3 fichiers sous format CSV. (*comma-separated-values*) qui est un format texte ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules. Dans notre cas, on a utilisés seulement deux fichiers.

Elliptic-txs-classes

Ce fichier contient deux colonnes, la première contient l'id de la transaction, et la deuxième colonne contient la classe associé à la transaction. Il existe deux classes codées par deux chiffres décimaux :

0 représente une classe licite.
1 représente une classe illicite.

Elliptic-txs-features

Ce fichier contient les transactions et leurs caractéristiques. Il ya 166 caractéristiques associées a chaque transaction.
Les 94 premières représente des informations locales sur la transaction y compris le pas de temps (représente une mesure du temps ou la transaction à été diffusée sur le réseau bitcoin), le nombre d'entrées/sorties, les frais de transaction, le volume de sortie et les chiffres agrégés tels que le BTC moyen reçu par les entrées et le BTC moyenne dépensé par les sorties, et le nombre moyen de transactions entrantes associées aux entrées et le nombre moyen de transactions sortantes associées aux sorties.
Les 72 caractéristiques restantes sont des fonctionnalités agrégées, obtenues en utilisant les informations de transaction, telle que le maximum, le minimum, l'écart type et les coefficients de corrélation des transactions voisins pour les même données d'informations (nombre d'entrées/sorties,frais de transactions, etc.)[50].

4.3.1 Partitionnement du jeu de données

Le jeu de données utilisé est composé de 203769 transactions. Deux pour cent (4545) des transactions sont étiquetées classe 1 (illicite). Vingt pour cent (42019) sont étiquetées classe 0 (licite). Les transactions restantes ne sont pas étiquetées en ce qui concerne les activités licites et illicites (*Unknown*).

Dans le cadre de notre travail nous avons seulement utilisés les transactions étiquetées. Nous avons divisés ces transactions en 3 parties.

- **Données d'entrainement :** Cette ensemble contient 3181 transactions illicites (soit 70% des transactions illicites), et leur équivalence en transactions licites (3181 transactions licites).
- **Données de validation :** Cette ensemble contient 908 transactions illicites (soit 20% des transactions illicites), et 908 transactions licites.
- **Données de test :** Cette ensemble contient 456 transactions illicites (les 10% restant), et les 37930 transactions licites restantes.

4.4 Mesure d'évaluation

4.4.1 Précision (*Accuracy*)

La précision est une méthode de mesure des performances d'un modèle de classification, exprimée généralement en pourcentage. La précision est le nombre de prédictions où la valeur prédite est égale à la valeur vraie. En d'autres termes, la précision est le rapport entre le nombre de prédictions correctes et le nombre total d'échantillons d'entrée, elle calcule le pourcentage de valeurs prédits qui correspondent aux valeurs réelles.

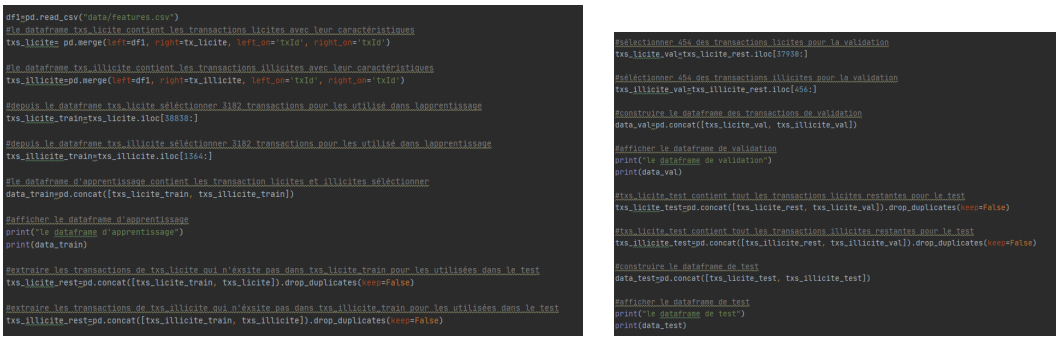
$$Accuracy = \frac{\text{Nombre de prédictions correctes}}{\text{Nombre de prédictions faites}}$$

4.4.2 Perte (*Loss*)

Une fonction de perte est une somme des erreurs commises pour chaque échantillon dans les ensembles de formation ou de validation. Elle prend en compte les probabilités ou l'incertitude d'une prédiction en fonction de l'écart entre la prédiction et la valeur réelle. Plus la perte est faible, plus nos prévision sont proches des vraies étiquettes.

4.5 Extrait du code utilisé

4.5.1 Construire les ensembles de données



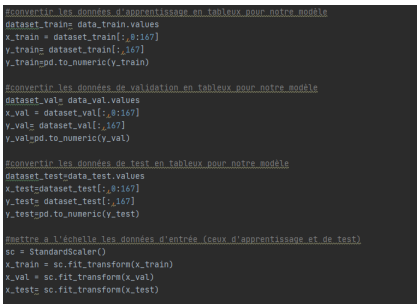
```
df1=pd.read_csv("data/features.csv")
#le dataframe txs_licite contient les transactions licites avec leur caractéristiques
txs_licite= pd.merge(left=df1, right=tx_licite, left_on='txId', right_on='txId')

#le dataframe txs_illicite contient les transactions illicites avec leur caractéristiques
txs_illicite=pd.merge(left=df1, right=tx_illicite, left_on='txId', right_on='txId')

#depuis le dataframe txs_licite séléctionner 3182 transactions pour les utilisé dans lapprentissage
txs_licite_train=txs_licite.iloc[38838:]

#depuis le dataframe txs_illicite séléctionner 3182 transactions pour les utilisé dans lapprentissage
txs_illicite_train=txs_illicite.iloc[1364:]

#le dataframe d'apprentissage contient les transaction licites et illicites séléctionner
data_train=pd.concat([txs_licite_train, txs_illicite_train])

#afficher le dataframe d'apprentissage
print("le dataframe d'apprentissage")
print(data_train)

#extraire les transactions de txs_licite qui n'éxsite pas dans txs_licite_train pour les utilisées dans le test
txs_licite_rest=pd.concat([txs_licite_train, txs_licite]).drop_duplicates(keep=False)

#extraire les transactions de txs_illicite qui n'éxsite pas dans txs_illicite_train pour les utilisées dans le test
txs_illicite_rest=pd.concat([txs_illicite_train, txs_illicite]).drop_duplicates(keep=False)
```



```
#séléctionner 454 des transactions licites pour la validation
txs_licite_val=txs_licite_rest.iloc[37938:]

#séléctionner 454 des transactions illicites pour la validation
txs_illicite_val=txs_illicite_rest.iloc[456:]

#construire le dataframe des transactions de validation
data_val=pd.concat([txs_licite_val, txs_illicite_val])

#afficher le dataframe de validation
print("le dataframe de validation")
print(data_val)

#txs_licite_test contient tout les transactions licites restantes pour le test
txs_licite_test=pd.concat([txs_licite_rest, txs_licite_val]).drop_duplicates(keep=False)

#txs_licite_test contient tout les transactions illicites restantes pour le test
txs_illicite_test=pd.concat([txs_illicite_rest, txs_illicite_val]).drop_duplicates(keep=False)

#construire le dataframe de test
data_test=pd.concat([txs_licite_test, txs_illicite_test])

#afficher le dataframe de test
print("le dataframe de test")
print(data_test)
```

FIGURE 4.9 – Extrait du code pour construire les ensembles des données utilisé.

4.5.2 Construire les modèles



```
#convertir les données d'apprentissage en tableux pour notre modèle
dataset_train= data_train.values
x_train = dataset_train[:,8:167]
y_train= dataset_train[:,167]
y_train=pd.to_numeric(y_train)

#convertir les données de validation en tableux pour notre modèle
dataset_val= data_val.values
x_val = dataset_val[:,8:167]
y_val= dataset_val[:,167]
y_val=pd.to_numeric(y_val)

#convertir les données de test en tableux pour notre modèle
dataset_test=data_test.values
x_test=dataset_test[:,8:167]
y_test= dataset_test[:,167]
y_test=pd.to_numeric(y_test)

#mettre a l'échelle les données d'entrée (ceux d'apprentissage et de test)
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_val = sc.fit_transform(x_val)
x_test= sc.fit_transform(x_test)
```

```
#configuration de l'architecture du réseau(modele fonctionnel)
inputs = keras.Input(shape=(167,))
dense = layers.Dense(85, activation="selu", kernel_initializer="uniform")
x = dense(inputs)
x=layers.Dropout(rate_=_0.1)(x)
outputs = layers.Dense(2, activation="softmax", kernel_initializer="uniform")(x)
model = keras.Model(inputs=inputs, outputs=outputs, name="FNCT1_model")
model.summary()
keras.utils.plot_model(model, "FNCT2_model.png")
model.compile(optimizer='rmsprop',loss_=_'sparse_categorical_crossentropy',metrics_=_["accuracy"])#ajouter la fonction compile (decent du gardiant)
hist=model.fit(x_train, y_train, batch_size_=_10, epochs_=8, verbose=1, validation_data=(x_val, y_val))#apprentissage du modèle

#commencé le test et voire le taux d'erreur et de précision
pred_train =model.predict(x_train)
scores = model.evaluate(x_train, y_train, verbose=1)
print('Accuracy on training data: {}% \n Error on training data: {}'.format(scores[1], 1 - scores[1]))

y_pred = model.predict(x_test)
scores2 = model.evaluate(x_test, y_test, verbose=1)
print('Accuracy on test data: {}% \n Error on test data: {}'.format(scores2[1], 1 - scores2[1]))
```

FIGURE 4.10 – Extrait du code pour construire les modèles en couche de classification.

4.5.3 Affichage des résultats

```
#afficher les graphiques de loss train
plt.plot(hist.history['loss'])
plt.title('Model loss')
plt.ylabel('loss')
plt.xlabel('Epoch')
plt.legend(['Train'])
plt.show()

plt.plot(hist.history['accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train'])
plt.show()

plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.ylabel('loss')
plt.xlabel('Epoch')
plt.legend(['val'])
plt.show()

plt.plot(hist.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['val'])
plt.show()
```

FIGURE 4.11 – Extrait du code pour afficher le résultat de l'entrainement et le test des modèles.

4.6 Entrainement des modèles et résultats obtenus

Les modèles décrit dans le chapitre précédent sont entrainés sur l'ensemble d'entrainement. L'ensemble de validation est utilisé pour valider les modèles. Puis, les modèles sont testés sur l'ensemble de test.
Dans ce qui suit les résultats obtenus pour chaque modèle.

4.6.1 Résultats obtenus pour le modèle 1

Nous représentons ci-dessus les courbes de précision et de perte obtenus lors de l'entrainement du modèle 1, et sa validation.

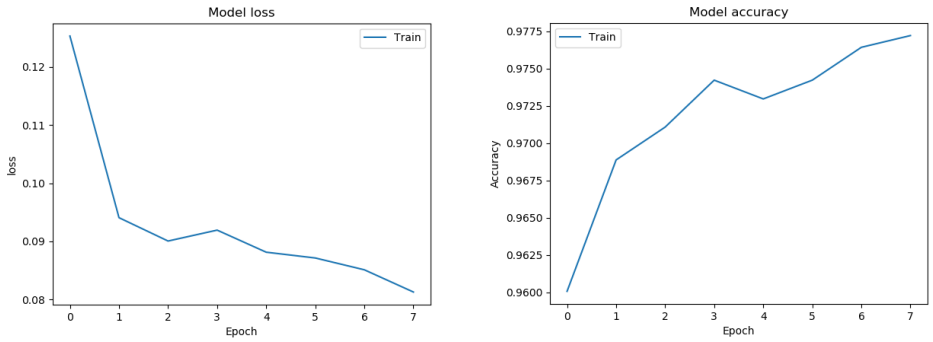


FIGURE 4.12 – Perte et précision du modèle 1 sur les données d'entraînement.

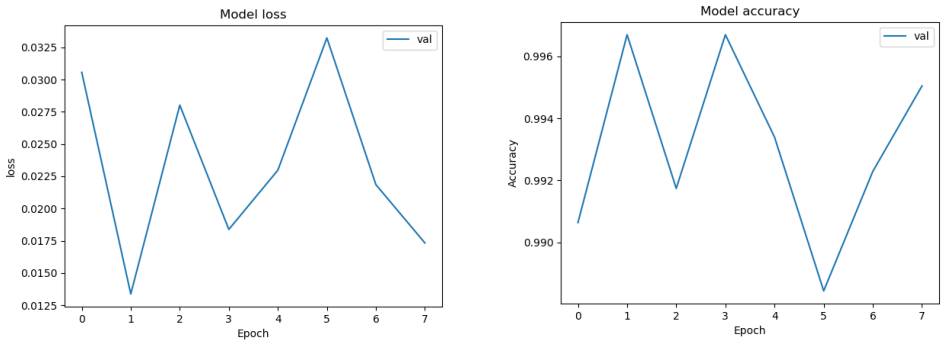


FIGURE 4.13 – Perte et précision du modèle 1 sur les données de validation.

Lors de l'entrainement (figure 4.9) , la précision (*accuracy*) augmente avec le nombre d'époques. Cependant la perte diminue. Ce qui reflète la capacité du modèle à apprendre.
Lors de la validation (figure 4.10), la perte diminue dans l'intervalle d'époques [0-1] et la précision augmente. Dans l'intervalle d'époques [1-5] la perte augmente et la précision diminue. Dans l'intervalle [5-10] la perte diminue et la précision augmente.
la figure suivante montre les résultats obtenus pour le modèle1.

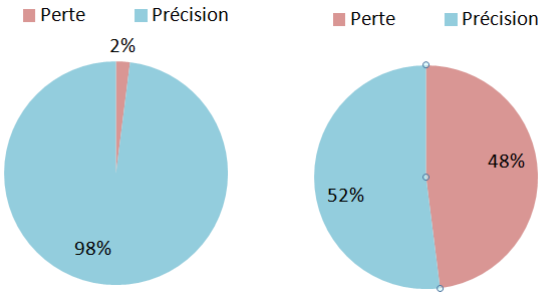


FIGURE 4.14 – Perte et précision finales sur l'ensemble d'entrainement et de test obtenue pour le modèle 1.

4.6.2 Résultats obtenus pour le modèle 2

Les courbes suivantes (figures 4.12 et 4.13) représentent la perte et la précision du modèle 2 sur l'ensemble d'entrainement et l'ensemble de test.

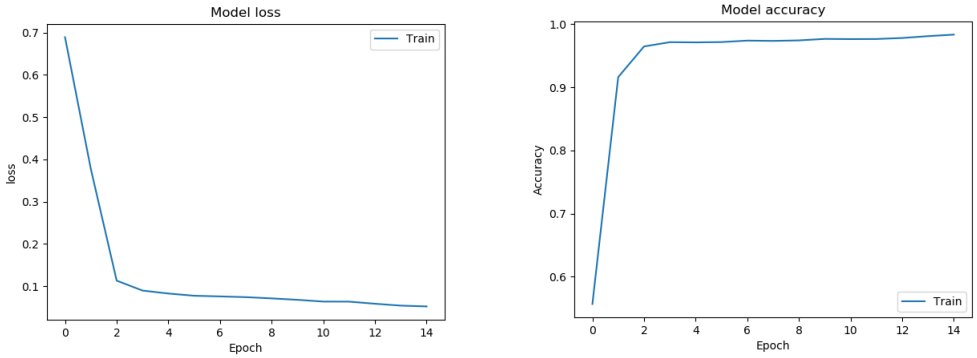


FIGURE 4.15 – Perte et précision du modèle 2 sur les données d'entraînement.

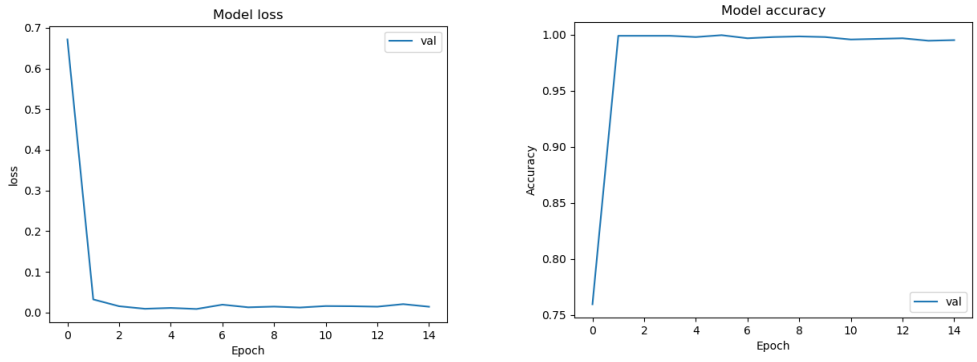


FIGURE 4.16 – Perte et précision du modèle 2 sur les données de validation.

Selon les figures 4.12 et 4.13, la précision de l'apprentissage et de validation augmente avec le nombre d'époques. De même, l'erreur d'apprentissage et de validation diminue avec le nombre d'époques.

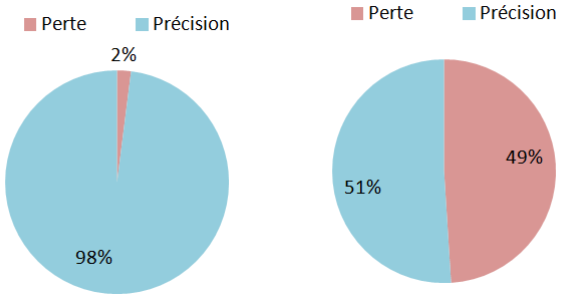


FIGURE 4.17 – Perte et précision finales sur l'ensemble d'entrainement et de test obtenues pour le modèle 2.

4.6.3 Résultats obtenus pour le modèle 3

Les figures qui suit représentent les courbes de précision et de perte obtenus par le troisième modèle lors de l'entrainement et de validation.

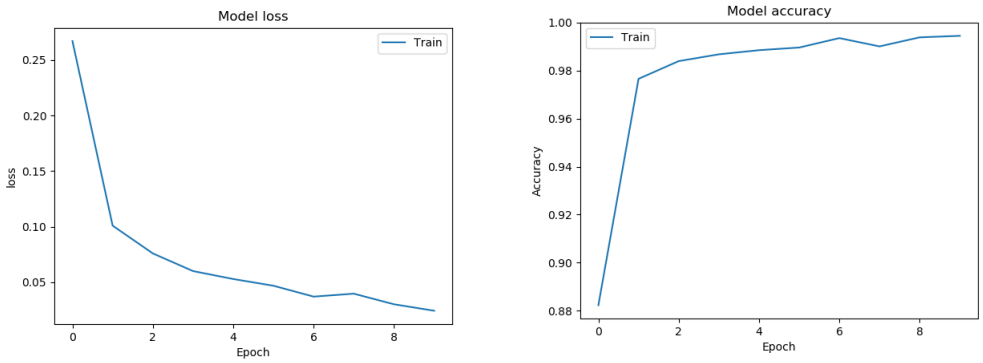


FIGURE 4.18 – Perte et précision du modèle 3 sur les données d'entraînement.

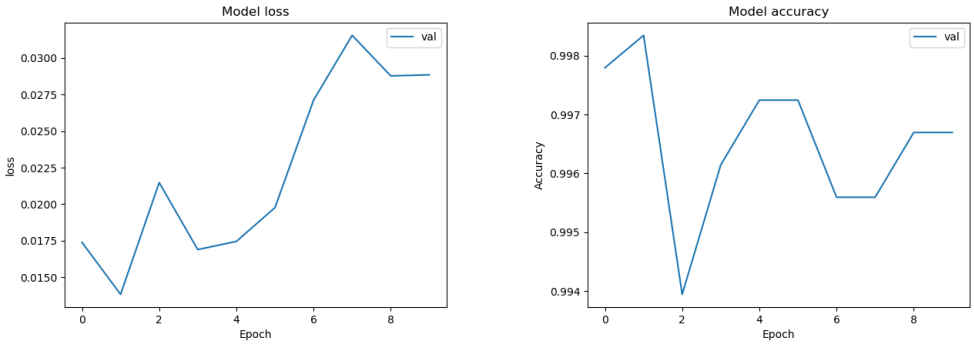


FIGURE 4.19 – Perte et précision du modèle 3 sur les données de validation.

Lors de l'apprentissage (figure 4.15), la perte diminue et la précision augmente avec le nombre d'époques.
Lors de la validation (figure 4.16), on remarque que la précision augmente et diminue au fil des itérations, jusqu'à atteindre 99,7% à l'époque 10, et de même pour la perte qui atteint 0.27% à l'époque10.

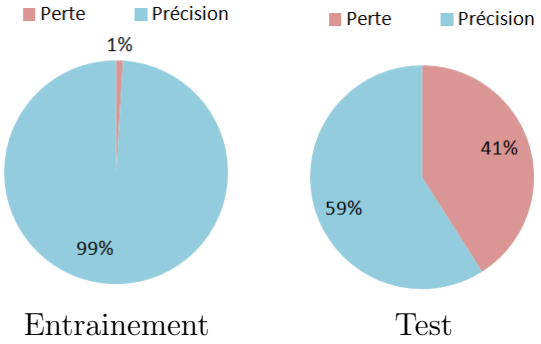


FIGURE 4.20 – Perte et précision finales sur l'ensemble d'entrainement et de test obtenues pour le modèle 3.

4.7 Discussion

Le tableau ci-dessous montre les différents résultats obtenus sur les trois modèles.

	Nombres d'époques	Précision finale sur l'ensemble d'entrainement	Précision finale sur l'ensemble de test	Temps d'exécution
Modèle 1	8	97%	51%	1 min 22
Modèle 2	15	97%	52%	1 min 18
Modèle 3	10	99%	59%	1 min 06

TABLE 4.1 – Tableau comparatif des résultats obtenus par les trois modèles.

Les résultats sont exprimés en termes de précision, erreur et enfin de temps d'exécution.
Les résultats obtenus par les modèles 1 et 2 sont très proches. Cependant, le troisième modèle a présenté de meilleurs résultats. ce qui signifie que l'ajout des couches cachées à notre réseau permet une certaine amélioration. Aussi le nombre d'époque joue un role important dans l'amélioration des résultats. Ces résultats obtenus peuvent être optimisés en approfondissant le réseau par l'ajout de couches cachées ou bien par l'augmentation du nombre d'époques.

4.8 Conclusion

Nous avons présentés dans ce chapitre les logiciels, librairies et configuration utilisés pour notre implémentation. Puis on a interprété les différents résultats obtenus en termes de précision, d'erreur et en temps d'exécution.

La comparaison des résultats trouvés à montré que le nombre d'époques, la profondeur du réseau sont des facteurs importants pour l'obtention de meilleurs résultats.

Conclusion générale

1 Synthèse

Dans ce mémoire nous avons présenté une approche de désanonymisation des clients dans le réseau *bitcoin* à l'aide de l'apprentissage automatique, plus particulièrement les réseaux de neurones artificiels. Nous avons commencé par l'apprentissage automatique et les réseaux de neurones artificiels. Par la suite, nous avons présenté la *blockchain bitcoin* et son fonctionnement. Nous avons passé ensuite à présenter notre approche pour la désanonymisation des transactions *bitcoins* (plus précisément leur classification en tant que transaction licite ou illicite), ainsi que les différents modèles d'apprentissage réalisés ainsi que leurs évaluation.

Bien que les résultats obtenus soient satisfaisants, un travail de recherche additionnel est nécessaire pour améliorer les résultats et procéder à d'autres formes de désanonymisation plus concrètes.

2 Perspective

Comme perspectives à ce projet, nous envisageons de :
- Intégrer l'apprentissage par renforcement aux modèles déjà réalisés.
- Améliorer les résultats obtenus en ajoutant des couches cachées aux différents modèles réalisés.
- Tester les modèles réalisés avec d'autres optimiseurs et hyper-paramètres.

Bibliographie

[1] la rédaction de futura. (2018) Qui sont les pionniers de l'intelligence artificielle ? [Online]. Available : https ://www.futura-sciences.com/tech/questions-reponses/intelligence-artificielle-sont-pionniers-intelligence-artificielle-4907/

[2] A. Cornuéjols, L. Miclet, and V. Barra, *Apprentissage artificiel deep learning concepts et algorithmes*. Eyrolles, 2018.

[3] A. Cornuéjols, L. Miclet, and Y. Kodratoff, *Apprentissage artificiel,concepts et algorithmes*. Eyrolles, 2002.

[4] H. Larochelle, "Étude de techniques d'apprentissage non supervisé pour l'amélioration de l'entrainement supervisé de modèles connexionnistes," Ph.D. dissertation, Université de Montréal, 2008.

[5] A. Cornuéjols and L. Miclet, *Apprentissage artificiel,concepts et algorithmes*. Eyrolles, 2010.

[6] K. Gosalia and R. Lefebvre, *Introduction à l'apprentissage automatique*. CPA New Brunswick, 2019.

[7] M. Taffar. (2018) Initaition a l'apprentissage automatique. [Online]. Available : https ://www.electronique-mixte.fr/formation-pdf/formation-pdf-intelligence-artificielle/cours-53-initiation-a-lapprentissage-automatique/

[8] E. Mathieu-Dupas. (2010) Algorithmes des k plus proches voisins. [Online]. Available : https ://hal.inria.fr/inria-00494814/document

[9] Y. Benzaki. (2017) Naive bayes classifier pour machine learning. [Online]. Available : https ://mrmint.fr/naive-bayes-classifier

[10] ——. (2018) Tout ce que vous voulez savoir sur l'algorithme k-means. [Online]. Available : https ://mrmint.Fr/algorithme K-means

[11] B. Kanber. (2013) Algorithme genetique. [Online]. Available : http ://igm.univ-mlv.fr/dr/XPOSE2013/tleroux-genetic-algorithm/fonctionnement

[12] P. Tirilly. (2006) Evaluation des performances des réseaux de neurone. [Online]. Available : http ://docplayer.fr/33710295-Evaluation-des-performances-des-reseaux-de-neurones-aleatoires-et-application-a-la-bio-informatique-tirilly-pierre-8-fevrier-2006.html

[13] la rédaction de wikistat. (2015) Les réseaux de neurones. [Online]. Available : https ://www.math.univ-toulouse.fr/ besse/Wikistat/pdf/st-m-app-rn.pdf

[14] G. Deyfus, "Les reseaux de neurones," 1998. [Online]. Available : https ://www.neurones.espci.fr/Articles-PS/GAMI.pdf

[15] Y. Djeriri. (2017) les réseaux de neurones artificiels. [Online]. Available : https ://www.academia.edu/37046526/Les-RC3A9seaux-de-Neurones-Artificiels

[16] A. Pottiez and M. Duquesnoy. (2018) Intelligence artificielle et apprentissage de réseaux connexionnistes. [Online]. Available : http ://math.univ-lille1.fr/calgaro/TER-2018/wa-files/duquesnoy-pottiez.pdf

[17] A. Salvail-Bérard. (2012) Réseaux de neurones. [Online]. Available : http ://docplayer.fr/84604798-Reseaux-de-neurones-adam-salvail-berard-6-septembre-2012.html

[18] A. Cornuéjols. (2007) Les réseaux de neurones. [Online]. Available : https ://www.lri.fr/antoine/Courses/ENSTA/Tr-ensta-nnx9.pdf

[19] l'équipe de U, *Comprendre la blockchain.* U.uchange, 2016.

[20] C. Jeanneau, A. Yeretzian, A. Stachtchenko, and C. Balva, *La blockchain décryptée, les clefs d'une révolution.* Observatoire Netexplo, 2016.

[21] V. F. Muntin, C. DeGanay, and R. Legleut. (2018) Les enjeux technologiques des blockchains (chaÎnes de blocs). [Online]. Available : http ://www.senat.fr/rap/r17-584/r17-5841.pdf

[22] l'équipe de bitcoin.fr. (2020) les grandes dates de bitcoin. [Online]. Available : https ://bitcoin.fr/histoire/

[23] C. Dorothe, *Blockchain, la révolution de l'économie de partage.* SMILE, 2017.

[24] P. Adam-Kalfon, *Blockchain, catalyseur de nouvelles approches en assurance.* PWC, 2018.

[25] crypto encyclopedie. (2017) Blockchain :qu'est ce qu'un mineur. [Online]. Available : https ://www.cryptoencyclopedie.com/single-post/Quest-ce-quun-mineur-blockchain-bitcoin

[26] P. Noizat. (2020) Ecdsa, technologie clé de bitcoin. [Online]. Available : https ://e-ducat.fr/links/ecdsa

[27] cipher. Sha256,sha384,sha512. [Online]. Available : http ://www.iwar.org.uk/comsec/resources/cipher/sha256-384-512.pdf

[28] L. Lars. (2019) Qu'est-ce qu'un bloc dans la technologie blockchain ? [Online]. Available : https ://cryptoast.fr/bloc-blockchain-crypto-explication/

[29] P. Pares. (2018) Learn the bitcoin system implementation with javascript code snippets for node.js. [Online]. Available : https ://pascalpares.appspot.ovh/an-introduction-to-the-bitcoin-system

[30] A. M. Antonopoulos. (2016) Mastering bitcoin. [Online]. Available : https ://bitcoin.fr/wp-content/uploads/2016/01/Mastering-Bitcoin.pdf

[31] youcoin.ch. (2015) Qu'est-ce qu'une preuve de travail (proof of work) ? [Online]. Available : http ://youcoin.ch/questions-reponses-faq/quest-ce-quune-preuve-de-travail-proof-of-work/

[32] P. Hooda. (2019) Proof of stake (pos) in blockchain. [Online]. Available : https ://www.geeksforgeeks.org/proof-of-stake-pos-in-blockchain/

[33] M. Laurent, *La Blockchain est-elle une technologie de confiance ?* Institut Mines-Télécom, 2017.

[34] équipe quantmetry. (2018) Blockchain : fonctionnement du protocole. [Online]. Available : https ://www.quantmetry.com/blockchain-fonctionnement-du-protocole

[35] V. Lachene, *Livre blanc sur la technologie blockchain.* Sigma Gestion, 2018.

[36] K. Sharad and G. Danezis. (2014) An automated social graph de-anonymization technique. [Online]. Available : https ://arxiv.org/abs/1408.1276

[37] Q. ShenTu and J. Yu. (2015) Transaction remote release (trr) : A new anonymization technology for bitcoin. [Online]. Available : https ://arxiv.org/abs/1509.06160

[38] A. SM and L. Av. (2018) Bitcoin user deanonimization methodes. [Online]. Available : https ://www.ispras.ru/proceedings/docs/2018/30/1/isp-30-2018-1-89.pdf

[39] R. Klusman and T. Dijkhuizen. (2018) Deanonymisation in ethereum using existing methods for bitcoin. [Online]. Available : https ://www.semanticscholar.org/paper/Deanonymisation-in-Ethereum-Using-Existing-Methods-Klusman/3bf10d4f747863996f0149b485ef8443da13d23a ?

[40] J. D. Nick. (2015) Data-driven de-anonymization in bitcoin. [Online]. Available : https ://jonasnick.github.io/papers/thesis.pdf

[41] G. karame and E. Androulaki, *Bitcoin and blockchain security.* ARTECH HOUSE, 2016.

[42] A. Biryukov, D. Khovratovich, and I. Pustogarov. (2014) Deanonymisation of clients in bitcoin p2p network. [Online]. Available : https ://arxiv.org/abs/1405.7418

[43] S. nakamoto. (2009) Bitcoin : a peer to peer electronic cash system. [Online]. Available : https ://bitcoin.org/bitcoin.pdf

[44] S. Meiklejohn, M. Pomarome, G. Jordan, K. Levchenko, D. Mcloy, G. M.Voelker, and S. Savage. (2013) A fistful of bitcoins : characterizing payment anonymen with no names. [Online]. Available : http ://www0.cs.ucl.ac.uk/staff/s.meiklejohn/files/login13.pdf

[45] D. Ron and A. Shamir. (2013) Quantitative analysis of the full bitcoin transaction graph. [Online]. Available : https ://eprint.iacr.org/2012/584.pdf

[46] F. Reid and M. Harrigan. (2012) An analysis of anonymity in the bitcoin system. [Online]. Available : https ://arxiv.org/abs/1107.4524

[47] P. Koshy, D. Koshy, and P. McDaniel, *An analysis of anonymity in the bitcoin using p2p network traffic.* Springer, 2014.

[48] M. moser, R. Bohme, and D. Breuker. (2013) An inquiry into money laundering tools in the bitcoin ecosystem. [Online]. Available : https ://maltemoeser.de/paper/money-laundering.pdf

[49] M. Moser, R. Bohme, and D. Breuker. (2014) Towards risk scoring of bitcoin transaction. [Online]. Available : https ://maltemoeser.de/paper/risk-scoring.pdf

[50] elliptic. (2019) Elliptic bitcoin dataset. [Online]. Available : https ://www.kaggle.com/ellipticco/elliptic-data-set/data