UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

VERS UNE DÉTECTION EFFICACE ET ROBUSTE DES FRAUDES BANCAIRES GRÂCE À LAPPRENTISSAGE AUTOMATIQUE

MÉMOIRE PRÉSENTÉ À L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE DE
LA MAÎTRISE EN MATHÉMATIQUES ET INFORMATIQUE APPLIQUÉES

PAR
CHEUN ANTHONY CEDRIC DA

OCTOBRE 2024

Résumé

Ces dernières années ont été marquées par une augmentation significative du volume et de la complexité des données relatives aux transactions financières. Cela a conduit à une recrudescence des activités frauduleuses. La détection des fraudes sur les cartes bancaires est donc devenue une préoccupation majeure. Ces fraudes sont de plus en plus complexes, évolutives et difficiles à détecter.

Ce mémoire de recherche est axé sur l'amélioration de l'efficacité et de la robustesse de la détection des fraudes bancaires via l'apprentissage automatique. Notre solution repose sur l'utilisation de plusieurs algorithmes évolués d'apprentissage automatique. Le déséquilibre des données demeure une problématique importante à résoudre dans le contexte de la détection des fraudes. Outre cela, nous tenterons de proposer une solution aux menaces alimentées par l'intelligence artificielle. Plusieurs buts de recherche sont liés à notre travail. Tout d'abord, nous allons étudier l'impact du déséquilibre des données sur la performance des algorithmes d'apprentissage automatique. Ensuite, le rééquilibrage des données ainsi que l'analyse de son impact sur la performance des algorithmes d'apprentissage automatique est également un objectif clé de notre travail. Enfin, ce travail a pour objectif l'étude des impacts des attaques alimentées par l'intelligence artificielle ainsi que l'élaboration d'une solution pour contrer ce type d'attaques qui sont de plus en plus récurrentes.

Remerciements

À mon père, ma mère, ainsi qu'a ma famille, je tiens à leur témoigner un profond remerciement pour toute l'aide qu'ils m'ont apporté.

À mes amis, merci pour tous les encouragements les conseils que j'ai reçus de votre part.

À ma directrice de recherche madame Hakima Ould-Slimane je vous remercie d'avoir accepté de m'encadrer ainsi que pour toutes les recommandations que vous m'avez prodiguées qui m'ont permis d'aboutir à ce travail.

Aux membres de jury, merci d'avoir accepté d'évaluer mon travail.

Merci à tous ceux que j'aurais oublié de citer et qui ont contribué de près ou de loin dans l'aboutissement de ce travail.

Glossaire

DerFAM : Détection efficace et robuste des fraudes via l'apprentissage automatique.

IA : Intelligence artificielle.

FGSM : Fast Gradient Sign Method.

PGD : Projected Gradient Descent.

(C & W) : Carlini and Wagner (C & W) attack.

Attaque DoS : Une attaque par déni de service (DoS) est une attaque visant à mettre hors service une machine ou un réseau, le rendant inaccessible aux utilisateurs prévus.

L'apprentissage fédéré : L'apprentissage fédéré est un sous-domaine de l'apprentissage automatique qui se concentre sur les situations dans lesquelles plusieurs entités forment un modèle en collaboration tout en veillant à ce que leurs données restent décentralisées.

Deeplocker : C'est une attaque malicieuse basée sur l'intelligence artificielle qui se déclenche uniquement quand la cible visée est atteinte.

CSV : C'est un type de fichier qui stocke des données tabulaires en texte brut, chaque ligne du fichier représentant généralement un enregistrement de données.

ACP : L'analyse en composantes principales est une méthode de réduction de la dimensionnalité et d'apprentissage automatique utilisée pour simplifier un grand ensemble de données en un ensemble plus petit tout en conservant des modèles et des tendances significatifs.

LSTM : Venant du terme anglais "Long short-term memory" , c'est un type de réseau neuronal récurrent qui vise à résoudre le problème de la disparition du gradient.

GRU : Venant du terme anglais "Gated Recurrent Unit (GRU)", il s'agit d'un réseau neuronal récurrent qui utilise moins de mémoire et plus rapide que le LSTM.

SPAM : Il est envoyé en masse , généralement envoyée par courrier électronique, le spam est également distribué par le biais de messages textuels (SMS), de médias sociaux ou d'appels téléphoniques.

Table des matières

Liste des tableaux

Table des figures

Chapitre 1

Introduction

Au cours des dernières années, nous avons assisté à une augmentation exponentielle des cas de fraudes financières [5]. Cette augmentation exponentielle s'explique en partie par la numérisation croissante de notre monde. En effet, de nos jours, il est courant de faire recours à des moyens de paiement en ligne que ce soit pour la facture d'eau, courant, téléphone, électricité, eau, etc [5]. Outre cela, les entreprises et les particuliers génèrent et stockent énormément de données sous forme électronique, ce qui les expose davantage à ce fléau. En conséquence, les fraudeurs ont intensifié leurs activités frauduleuses, en mettant en place plusieurs stratagèmes touchant plusieurs secteurs financiers, comme la banque, l'assurance, la fiscalité. Non seulement les fraudes sont de plus en plus nombreuses, mais également elles sont de plus en plus complexes à détecter [21]. Ce fléau a pris une telle ampleur qu'en 2022 les pertes globales sur les cartes de crédit ont été estimées à environ 22 milliards de dollars américains avec des pertes mondiales prévues atteignant le chiffre stupéfiant de 43 milliards de dollars d'ici 2026 [25]. Cela souligne donc la nécessité d'une détection efficace des fraudes.

Le volume est colossal, il est quasi impossible pour un humain de les analyser et de pouvoir détecter les schémas de fraude. Cela a donc conduit les chercheurs à s'orienter vers des techniques plus modernes, comme l'apprentissage automatique. Ce choix s'explique par la capacité des algorithmes d'apprentissage automatique à analyser et à identifier des schémas de fraude dans des quantités de données gigantesques [59],[69]. Cependant la confidentialité des transactions bancaires demeure un frein à l'apprentissage des algorithmes automatique, car les données sont difficiles d'accès, outre cela ce type de données étant fortement déséquilibré, cela affecte la performance des algorithmes d'apprentissage automatique [39],[10],[20],[12].

Plusieurs objectifs sont liés à notre travail de recherche. Tout d'abord, nous allons étudier l'impact qu'a le déséquilibre des données sur la performance des algorithmes d'apprentissage automatique. Ensuite, nous allons procéder à la résolution du déséquilibre présent dans nos données et nous effectuerons de nouveau des tests de

performance. Ces nouveaux tests de performance auront pour but de vérifier si nous notons une amélioration de performance des algorithmes d'apprentissage automatique après le rééquilibrage des données. Enfin, l'amélioration de la robustesse de notre modèle contre les attaques adverses demeure un objectif clé de notre travail de recherche. Dans un premier temps, nous allons étudier l'impact qu'a l'attaque adverse ZOO(zeroth order optimisation) sur la performance de notre système. Par la suite, nous allons proposer une solution efficace pour contrer cette attaque.

Dans ce travail de recherche, notre principale contribution est l'amélioration de l'efficacité de la détection des fraudes ainsi que la protection des systèmes de détection des fraudes contre les attaques adverses. En effet, compte tenu de la numérisation croissante dans notre monde, les fraudes sont amenées à se multiplier davantage. Également avec l'avancée de l'intelligence artificielle, les fraudeurs font de plus en plus recours à des attaques alimentées par l'intelligence artificielle pour tromper les systèmes de détection des fraudes, d'où la nécessité d'une meilleure protection des systèmes contre ce type d'attaque. Notre travail de recherche peut être subdivisé en deux grandes parties. La première partie consistera à la résolution du déséquilibre présent dans les données de transactions financières. Ainsi donc nous utiliserons et nous comparerons plusieurs techniques de génération de données synthétiques [12],[31],[35],[53] afin de choisir celle qui convient le mieux dans notre contexte. Après avoir rééquilibré nos données, nous testerons la performance de plusieurs algorithmes d'apprentissage automatique. La seconde étape vise à l'amélioration de la robustesse de nos différents modèles d'apprentissage automatique, pour cela nous implémenterons une attaque adverse de type ZOO (zeroth order optimisation).

Ce mémoire est composé de six chapitres. Dans un premier temps, dans le premier chapitre, nous présenterons la problématique et les objectifs de recherche de ce mémoire. Le deuxième chapitre abordera la détection des fraudes ainsi que les algorithmes d'apprentissage automatique utilisés pour réaliser cette tâche. Le troisième chapitre, intitulé travaux connexes, présentera les différentes solutions élaborées dans les articles de recherche permettant la détection des fraudes. Dans le chapitre quatre, nous aborderons les techniques d'exploration des données en présentant de façons détaillées les concepts liés à ce domaine ainsi que les différentes étapes se rapportant à ce processus. Le chapitre cinq présentera à la fois la méthodologie de notre solution ainsi que les résultats et les discussions de nos différentes expérimentations. Le dernier chapitre à savoir la conclusion donnera une vue globale et synthétique sur l'ensemble de notre travail.

Chapitre 2

Notions et Concepts préliminaires

2.1 Introduction

Ces dernières années ont été marquées par une augmentation significative du volume ainsi que de la complexité des données de transactions financières [5]. En effet, les entreprises et les particuliers génèrent et stockent énormément de données sous forme électronique [4]. Cela a conduit à une recrudescence des activités frauduleuses. La détection des fraudes est donc devenue une préoccupation majeure à la fois pour les entreprises et les particuliers qui en sont de plus en plus victimes. L'augmentation des données frauduleuses souligne le besoin crucial d'une détection efficace de la fraude. Ces résultats sont également confirmés par l'étude menée par le Centre antifraude du Canada en 2022 dans laquelle il a été enregistré plus de 91190 signalements de fraude, également il ressort que 57055 victimes ont subi des pertes supérieures à 531 millions de dollars [18]. À cela, il nous faut ajouter les organismes ainsi que les particuliers ayant subi des fraudes sans les déclarer [18]. Ce fléau est mieux illustré sur la figure 2.1, qui présente les fraudes déclarées à la police de 2011 à 2021, sur cette figure nous constatons une hausse croissante des fraudes au fur et à mesure des années.

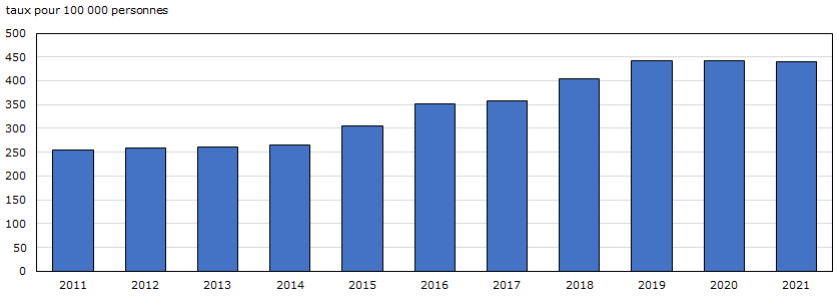
Avant d'aborder la question de la détection des fraudes, il est essentiel que nous comprenions ce qu'est une fraude. Si nous ne faisons pas cet effort, il sera impossible pour nous de pouvoir en identifier les manifestations. Bien qu'il existe plusieurs définitions d'une fraude , la façon la plus simple par laquelle nous pouvons définir une fraude est la suivante, une fraude est un acte de tromperie intentionnel ou de malhonnêteté perpétré par un ou plusieurs individus, généralement en vue d'un gain financier [4]. Pour pouvoir prouver une fraude il est nécessaire de pouvoir identifier les éléments suivants :

- La déclaration doit être fausse.
- L'individu doit savoir que la déclaration est fausse.
- L'intention de tromper la victime.

- ❖ La victime s'est fiée à la déclaration.
- ❖ La victime subit un préjudice financier ou autre.

L'élément majeur qui ressort d'une fraude demeure le côté intentionnel, c'est la tromperie intentionnelle qui incite la victime à adopter une ligne de conduite aboutissant à une perte qui distingue l'acte de vol [5]. Une simple erreur n'est pas frauduleuse si elle n'est pas commise dans le but d'induire la victime en erreur. Il est également important pour nous de comprendre la nuance existant entre une fraude et un abus, en effet on parle d'abus lorsque les éléments permettant d'identifier une fraude n'existent pas explicitement. Les exemples ci-dessous illustre les cas d'abus :

- ❖ Accéder à des sites Internet tels que Facebook et eBay pour des raisons personnelles.
- ❖ Prendre un congé de maladie alors que l'on n'est pas malade.
- ❖ Passer des appels téléphoniques personnels.
- ❖ Des performances délibérément insuffisantes.
- ❖ Prendre des fournitures de bureau à des fins personnelles.



Note : Les 168 483 affaires de fraude déclarées par la police en 2021 représentaient un taux de 441 affaires pour 100 000 personnes, soit un taux en légère baisse par rapport à l'année précédente (443), mais près de deux fois plus élevé qu'en 2011 (254). Parmi les affaires de fraude déclarées par la police, 14% étaient des fraudes d'identité et 4% étaient des vols d'identité. Les affaires d'extorsion, qui peuvent être considérées comme une forme de fraude bien qu'elles soient catégorisées en tant que crimes violents, ont aussi enregistré une hausse importante depuis 2011. De 2011 à 2021, le taux d'affaires d'extorsion déclarées par la police a quadruplé pour passer de 4,4 affaires à 17,6 affaires pour 100 000 personnes.

FIGURE 2.1 – Affaires de fraude déclarées par la police, Canada, 2011 à 2021. Source : https ://www150.statcan.gc.ca/n1/pub/89-652-x/89-652-x2023001-fra.htm

2.2 Anomalies Vs Fraudes

Dans le domaine de l'analyse des données, détecter une fraude peut être perçu comme chercher une aiguille dans une botte de foin, cela illustre donc la complexité de ce processus. Généralement, les transactions frauduleuses dans les enregistrements électroniques sont peu nombreuses par rapport à la grande quantité d'enregistrements

dans les ensembles de données [15],[32]. La détection des anomalies consiste à identifier des événements, des éléments ou des observations rares qui sont suspects parce qu'ils diffèrent de manière significative des comportements ou des modèles standards. Le tableau 2.1 illustre la comparaison entre anomalie et fraude. La détection des anomalies est une partie cruciale qui s'appuie sur des modèles statistiques pour identifier les schémas qui s'écartent de manière significative du comportement attendu. L'analyse comportementale améliore encore la détection des fraudes en évaluant les habitudes des clients et en identifiant les écarts susceptibles d'indiquer des actions frauduleuses.

De façon plus simple, la détection de fraude se concentre sur la détection d'activités malicieuses, quant à la détection d'anomalies, elle vise à identifier tout écart par rapport aux résultats normaux, qu'il soit le résultat d'une activité frauduleuse ou non. La détection d'anomalies a également d'autres domaines d'application notamment celui de la santé, en effet dans le domaine de la santé, elle peut permettre l'identification des cellules cancéreuses. [10].

Type	**Objectif**	**Se focalise**	**Domaine d'application**
Fraude	Identifier les activités malveillantes	Identifier et prévenir les activités frauduleuses	Finance
Anomalie	Identifier tout écart par rapport aux résultats normaux	Champ d'application plus large	Vaste(santé,finance, intrusions,..)

TABLE 2.1 – Tableau comparatif entre anomalie et fraude

2.3 Types de fraude

2.3.1 Fraudes à l'assurance

La fraude à l'assurance désigne tout acte commis dans le but de frauder un processus d'assurance. Elle se produit lorsqu'un demandeur tente d'obtenir une prestation ou un avantage auquel il n'a pas droit, ou lorsqu'un assureur refuse a un client une prestation qui lui est due [39]. Les fraudes à l'assurance sont très diverses et peuvent être regroupées en trois grands types de fraudes qui sont les suivantes :

- Fraude à l'assurance maladie : Elle peut être définie comme étant un acte intentionnel de tromperie, de dissimulation ou de fausse déclaration résultant par

un gain de prestation de santé non mérité pouvant être attribué à un individu ou à un groupe. Cette problématique est d'actualité et impacte fortement les programmes financés par les contribuables. Les exemples illustrant ce fléau sont les doubles facturations effectuées par certains médecins ou encore des chirurgiens qui pratiquent des interventions chirurgicales inutiles.

- ❖ Fraude à l'assurance automobile : C'est un type de fraude très fréquent, dans ce type de fraude un ou plusieurs fraudeurs peuvent simuler des décès sur la route ou mettre en scène des collisions afin de présenter des fausses demandes d'indemnisation.

- ❖ Fraude immobilière : Ce type de fraude peut résulter par des pertes énormes dont notamment la perte de votre propriété immobilière, également dans ce type de fraude, il est courant de constater sur votre dossier de crédit que des hypothèques supplémentaires ont été contractées à votre nom.

2.3.2 Fraudes bancaires

Les fraudes bancaires représentent un type de fraude qui a connu une augmentation importante au cours des dernières années, s'expliquant par la numérisation croissante dans le monde actuel [39],[5],[18]. Bien qu'existant plusieurs définitions de ce qu'est une fraude bancaire, de façon plus simple elle consiste en l'utilisation de moyens potentiellement illégaux pour obtenir de l'argent, des actifs ou d'autres biens appartenant ou détenus par une institution financière, ou pour obtenir de l'argent de déposant en se faisant passer frauduleusement pour une banque ou une autre institution financière [39]. Les différents types de fraudes bancaires sont les suivantes :

- ❖ Blanchiment d'argent : C'est un processus qui consiste à dissimuler illégalement l'origine de l'argent, obtenu par des activités illicites telles que le trafic de drogue, la corruption, le détournement de fonds ou les jeux d'argent, en le convertissant en une source légitime.

- ❖ Fraudes bancaires mobiles : C'est un phénomène qui a pris de l'ampleur avec l'apparition des téléphones intelligents qui sont de plus en plus utilisés pour les services financiers. Dans ce type de fraude, les fraudeurs imitent les appareils existants de la victime ou dans certains cas, ils simulent l'utilisation par la victime d'un nouvel appareil pour accéder à son compte bancaire. Pour que cette attaque soit efficace, il est nécessaire que le fraudeur ait accès aux identifiants de sa victime qui peut être obtenu par l'installation d'un logiciel malveillant présent sur l'appareil de sa victime. D'autres pistes suggèrent également que les fraudeurs peuvent avoir accès aux identifiants des victimes via le Dark Web.

- ❖ Fraude à la carte de crédit : C'est un type de fraude commise en utilisant les paiements par carte pouvant être des cartes de crédit ou des cartes de débit. Dans ce type de fraude, le fraudeur tente d'obtenir des biens ou des services ou d'effectuer un paiement sur un autre compte qu'il contrôle. La fraude à la carte de crédit peut être autorisée, lorsque le véritable client effectue lui-même le paiement sur un autre compte contrôlé par un criminel, ou non autorisé, lorsque le titulaire du compte ne donne pas l'autorisation d'effectuer le paiement et que la transaction est effectuée par un tiers.

2.3.3 Fraudes fiscales

Les fraudes fiscales constituent un type de fraude caractérisé par la volonté délibérée et intentionnelle d'un particulier ou une entité commerciale d'induire une entreprise en erreur sur le contenu de sa déclaration de revenus pour minimiser son imposition [39]. Dans ce type de fraude l'objectif principal est d'éviter de payer l'intégralité de l'obligation fiscale, pour cela le fraudeur peut utiliser plusieurs stratagèmes comme la demande de fausses déductions, la déclaration des dépenses personnelles comme étant des dépenses professionnelles, l'utilisation d'un faux numéro de sécurité sociale.

2.3.4 Fraudes d'entreprise

Les fraudes en entreprise désigne toute action illégale, contraire à l'éthique et trompeuse, commise soit par une entreprise, soit par un individu agissant en sa qualité d'employé de l'entreprise [39]. Ce type de fraude est particulièrement difficile à identifier et exige souvent un audit approfondi des finances de l'entreprise pour la détecter.

Dans ce type de fraude, le montant des fraudes est souvent très important et pouvant se chiffrer en milliard de dollars lorsqu'elles sont perpétrées par des cadres supérieurs d'une grande entreprise (multinationales). Les victimes de la fraude d'entreprise sont les consommateurs ou les clients, les créanciers, les investisseurs, les autres entreprises et, enfin, l'entreprise à l'origine de la fraude et ses employés. Lorsqu'elle est finalement découverte, l'entreprise qui a commis la fraude est souvent laissée à l'abandon et contrainte de déclarer faillite.

A l'image de tous les autres types de fraude le but principal recherché est un gain financier, cependant ce type de fraude peut être également lié à d'autres raisons dont notamment le désir ou le besoin perçu d'attirer ou de retenir les investisseurs. En effet, de nombreuses fraudes d'entreprises consistent en des montages comptables frauduleux utilisés pour faire apparaître une entreprise plus rentable qu'elle ne l'est en réalité et cela dans le but d'attirer ou de retenir les investisseurs. Une autre cause de ce type de

fraude est la volonté de l'entreprise ou d'un employé de dissimuler des problèmes ou des défauts dans leurs produits. Cela est souvent fréquent chez les sociétés pharmaceutiques, qui souhaitent cacher certains effets secondaires ou dangers liés à l'utilisation de certains médicaments qu'elles fabriquent et vendent.

2.3.5 Fraudes hypothécaires

La fraude hypothécaire désigne tout acte de tromperie ou de fausse déclaration commise dans le but d'obtenir un crédit hypothécaire [39]. En règle générale, il y a fraude hypothécaire lorsqu'un candidat à l'achat d'un logement donne de fausses informations ou omet des informations importantes dans le cadre d'une demande de prêt hypothécaire pour l'achat d'un bien immobilier. Il existe de multiples formes de fraude hypothécaire, parmi ces fraudes les plus courantes sont la fraude au revenu, la fraude à l'évaluation et la fraude à l'occupation.

Dans ce type de fraude, les fraudeurs sont généralement les emprunteurs, les courtiers ou même des prêteurs hypothécaires peu scrupuleux. Nous distinguons plusieurs exemples de fraude à l'hypothèque dont les plus courants sont les suivants :

- La fraude au revenu est le type de fraude hypothécaire le plus fréquent. Elle consiste pour un éventuel emprunteur à déclarer un montant de revenus nettement supérieur à celui qu'il perçoit réellement. Le développement de la criminalité sur Internet facilite cette pratique, car il est possible de trouver sur des sites Web payants des services comme une fausse vérification des revenus ou des fausses déclarations d'impôt. Parmi les autres méthodes de fraude que l'emprunteur peut utiliser, on peut citer la déclaration d'un revenu d'auto-emploi fictif ou l'indication d'un faux titre de travail. Dans des cas extrêmes, cela peut même aller jusqu'au vol d'identité afin d'utiliser les informations financières d'une autre personne pour obtenir un crédit hypothécaire et acheter un bien immobilier.

- On parle de fraude à l'évaluation lorsque l'évaluation d'un bien immobilier pour lequel un emprunteur souhaite obtenir un prêt hypothécaire est intentionnellement surévaluée ou sous-évaluée. Pour commettre une fraude à l'évaluation, un emprunteur potentiel doit généralement être de connivence avec un évaluateur malhonnête. Un exemple par lequel, nous pouvons illustrer ce cas est celui d'un emprunteur faisant appel à un évaluateur qui évalue un bien immobilier à un prix supérieur à sa valeur réelle ou à son prix de vente.

- La fraude à l'occupation peut se produire de différentes manières. Un emprunteur potentiel peut déclarer qu'il a l'intention d'occuper un bien immobilier en tant que résidence principale afin d'obtenir les conditions de prêt les plus favorables,

alors qu'il n'a pas l'intention d'y vivre. Les prêts sur les résidences principales sont généralement accordés à des taux d'intérêt plus bas. Un autre type de fraude à l'occupation se produit lorsqu'un emprunteur prétend faussement qu'il achète un bien d'investissement qu'il a l'intention de louer, en utilisant les revenus locatifs prévus pour l'aider à se qualifier pour un prêt hypothécaire

2.4 Évaluation des risques de fraude

Il est quasiment impossible de pouvoir éliminer entièrement le risque de fraude, cependant il est possible de pouvoir le minimiser. Dans cette optique, il est important que nous soyons à jour sur les différentes techniques et stratagèmes pouvant être utilisés par les fraudeurs, surtout avec la numérisation croissante. En effet, avec cette numérisation croissante, nous avons vu apparaître des stratagèmes vicieux utilisés par les fraudeurs changeants des attaques traditionnelles d'hameçonnage connues. L'exemple concret pouvant illustrer ce fléau est celui des attaques adverses, dans ce type d'attaque le fraudeur fait usage de l'intelligence artificielle en générant des données de fraude qu'il fera passer par la suite pour des données normales. Suite à cela le fraudeur enverra ces données au modèle de détection de fraude dans le but de tromper le modèle comme l'illustre la figure 2.2 [21].

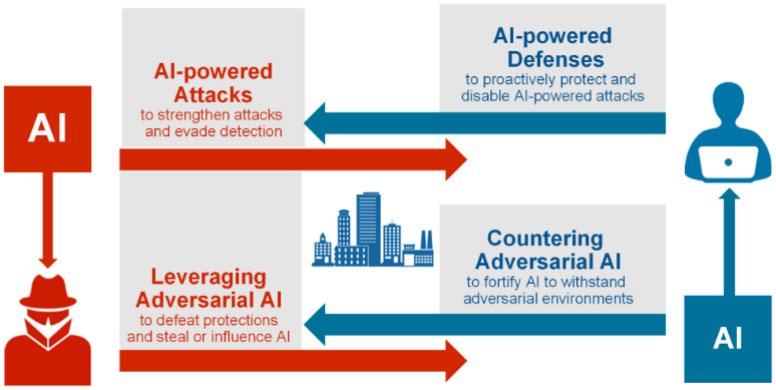


FIGURE 2.2 – Le double problème de l'intelligence artificielle : Source : Joysula Rao, IBM Corporation, presentation to the workshop, December 11, 2018

2.5 Détection des fraudes

La détection des fraudes désigne tout processus de surveillance des transactions et du comportement des clients afin de repérer et de combattre les activités frauduleuses. La vaste prolifération des données et les complexités technologiques croissantes ont rendu difficile la détection des fraudes. En effet, ayant des données de plus en plus complexes (structurées, semi-structurées et non structurées) et hétérogènes (provenant de diverses sources), les techniques d'analyse classiques des données reposant sur l'expertise

purement humaine sont devenues obsolètes. Les fraudeurs sont de plus en plus ingénieux et ont recours à de nouvelles technologies pour lancer des attaques sophistiquées et à grande échelle [5],[66]. Parmi ces technologies, nous avons l'intelligence artificielle dont l'avènement a rendu les attaques de plus en plus intelligentes et difficiles à détecter. En effet, plusieurs fraudeurs font usage de l'intelligence artificielle pour lancer des attaques complexes comme le Deeplocker, qui est un type de logiciel malveillant alimenté par l'intelligence artificielle et qui échappe à la détection de la plupart des contrôles de sécurité [21].

Les techniques de détection des fraudes doivent se concentrer sur la recherche d'anomalies dans les données, ainsi l'identification de schémas non usuels dans les données peuvent être de bons indicateurs de tentative de fraude. L'analyse des données permet donc l'extraction des informations utiles présentes dans les données pouvant soutenir la prise de décision [61],[10]. Plusieurs techniques d'analyse statistique ainsi que l'utilisation des multiples algorithmes d'apprentissage automatique ou d'apprentissage profond (réseaux de neurones) peuvent nous aider dans cette tâche. Les différentes techniques de statistique ainsi que les algorithmes d'apprentissage automatique seront amplement détaillés dans la suite de notre document. Les anomalies auxquelles nous pouvons nous intéresser lors de la détection des fraudes sont les suivantes :

- ❖ Des valeurs aberrantes là où elles ne sont pas attendues.
- ❖ Trop ou pas assez de transactions.
- ❖ Relations inhabituelles entre les éléments.
- ❖ Moment inattendu des transactions ou des événements.
- ❖ Comptes ou soldes de comptes inhabituels.
- ❖ Incohérences.
- ❖ Présences de doublons.
- ❖ Méthodes de paiement inattendues.

Parmi les défis se rapportant à la détection des fraudes, nous avons le déséquilibre des données financières, à cela nous devons ajouter l'apparition d'attaques modernes à base d'intelligence artificielle dont le but principal est d'induire en erreur les modèles de détection de fraude. L'utilisation des techniques d'exploration des données, des réseaux de neurones ainsi que des techniques d'analyse statistique avancées peuvent nous permettre d'identifier les irrégularités (anomalies) présents dans nos données. Plusieurs algorithmes de génération de données synthétiques peuvent être utilisés dans le but d'améliorer la performance des modèles de détection. Les attaques utilisant des logiciels malveillants alimentés par l'intelligence sont amenées à se multiplier dans les années à venir, il est donc nécessaire de pouvoir anticiper ces futures menaces. Plusieurs solutions peuvent être proposées pour contrer ces futures attaques, les plus pertinentes sont les

suivantes :

- ❖ Extraction de caractéristiques et reconnaissance des patterns pour améliorer la prise de décision et détecter les menaces inconnues.

- ❖ Traitement du langage naturel pour collecter du texte sur les violations passées et actuelles, consolider les informations sur les menaces et améliorer les connaissances en matière de sécurité.

- ❖ Automatisation des tâches pour réduire la charge de travail de l'analyste humain et diminuer le temps de réaction.

- ❖ Améliorer la modélisation des comportements des futurs modèles pour mieux identifier les menaces et les risques émergents et passés.

- ❖ Consolidation des sources de renseignements, y compris les sources de données structurées et non structurées.

2.6 Ensembles de données sur la fraude

En raison de la nature des données financières, il y a un manque accru d'ensemble de données accessibles au public qui peuvent être utilisées pour l'entraînement des modèles d'apprentissage automatique [39]. Outre le problème de manque de données, nous devons noter que les fraudes sont de plus en plus évolutives et subtiles au fil du temps [21]. Les données représentant les fraudes étant très minimes dans les ensembles de données, cela rajoute encore plus de complexité dans le processus d'analyse ainsi que celui de détection des fraudes. Le tableau 2.2 présente les ensembles de données de fraude les plus pertinents ayant retenu notre attention :

- ❖ Counter Fraud : C'est un ensemble de données produit par le Lincolnshire County Council, répertoriant des cas de fraude commis par des employés au cours d'une année. Cet ensemble est composé de 26 colonnes dont les plus significatifs qui répertorient les cas de fraude enregistrés dans le compte des employés, ainsi qu'une estimation du montant de fraude et du nombre d'investigations menées, du montant ayant pu être récupéré suite aux investigations. Ces informations sont publiées dans le cadre du Code de transparence du gouvernement local, il est également important de mentionner qu'un signalement de fraude peut être également indiqué pouvant déboucher à une nouvelle enquête. Cet ensemble de données est mis à jour chaque année au mois de juin, cependant la dernière mise

à jour date de 2019 [26].

- ❖ Ensemble de données synthétiques pour la détection des fraudes: Cet ensemble de données a été mis en place pour pallier au manque d'accessibilité de façon publique des données de transactions financières. Comme nous le savons, les données financières sont importantes pour de nombreux chercheurs et en particulier pour les chercheurs qui effectuent des recherches dans les domaines financiers, pour cela le simulateur Paysim nous permet de générer des données de transactions financières pour permettre un meilleur entraînement des modèles de détection de fraude [26].
- ❖ Credit Card Fraud Detection effectuées par cartes de crédit en septembre 2013 par des détenteurs de cartes européens [26]. Cet ensemble de données présente des transactions qui ont eu lieu en deux jours, où nous avons 492 fraudes sur 284 807 transactions. L'ensemble de données est fortement déséquilibré, la classe positive (fraudes) représentant 0,172% de toutes les transactions. Il ne contient que des variables d'entrée numériques qui sont le résultat d'une transformation ACP. Malheureusement, pour des raisons de confidentialité, nous ne pouvons pas fournir les caractéristiques originales et d'autres informations générales sur les données. Les caractéristiques V1, V2, ... V28 sont les composantes principales obtenues avec l'ACP. Les seules caractéristiques qui n'ont pas été transformées avec l'ACP sont "Temps" et "Montant". La caractéristique "Temps" contient les secondes écoulées entre chaque transaction et la première transaction de l'ensemble de données. La caractéristique "Montant" est le montant de la transaction, cette caractéristique peut être utilisée pour l'apprentissage sensible aux coûts en fonction de l'exemple. La caractéristique "Classe" est la variable de réponse et prend la valeur 1 en cas de fraude et 0 dans le cas contraire.

- ❖ Transactions IBM pour la lutte contre le blanchiment d'argent: La détection du blanchiment est très difficile. La plupart des algorithmes automatisés ont un taux élevé de faux positifs : des transactions légitimes sont incorrectement signalées comme étant des opérations de blanchiment. L'inverse est également un problème majeur : les faux négatifs, c'est-à-dire les transactions de blanchiment non détectées. L'accès aux données des transactions financières réelles est très limité, pour des raisons de propriété et de respect de la vie privée. Même lorsqu'il est possible d'y accéder, il est difficile d'attribuer une étiquette correcte (blanchiment ou légitime) à chaque transaction, comme nous l'avons vu plus haut.

Jeu de donnée	**type**	**Prétraitement**	**Lignes**	**caractéristiques**
Counter Fraud	Fraude des employés	Non	7	26 colonnes
Synthetic Fraud	Fraude	Non	10217	17 colonnes
Credit Card Fraud Detection	Fraude	ACP	284807	31 colonnes
Transactions IBM	Blanchiment d'argent	Non	5078345	11 colonnes

TABLE 2.2 – Tableaux présentant les ensembles de données.

2.7 Concepts préliminaires

La détection des anomalies est l'un des cas d'utilisation les plus courants de l'apprentissage automatique. En effet, la détection et l'identification des valeurs aberrantes permettent de prévenir les fraudes, les attaques adverses et les intrusions dans le réseau [39].

Avant de chercher à comprendre le concept d'anomalie, il nous faut comprendre d'abord ce que sont les valeurs aberrantes. Une valeur aberrante est un point de données différant fortement des valeurs normales, l'exemple le plus simple pouvant illustrer le concept de valeur aberrante c'est les Jeux olympiques de 2008 (tableau 2.3), dans lesquels Usain Bolt a établi un record du monde dans le 200 mètres hommes avec un temps de 19.30 secondes, terminant plus de six dixièmes de seconde plus vite que Shawn Crawford deuxième [39],[78]. Les cinq autres concurrents ont réalisé des temps compris entre 19.96 secondes et 20.59 secondes, ce qui signifie que la différence entre la deuxième et la dernière place était légèrement supérieure à la différence entre la première et la deuxième place. On comprend alors que les résultats de la deuxième à la sixième place sont donc des valeurs qui ne diffèrent pas significativement de la norme, ce qui n'est pas le cas de la première place. En analysant le tableau 2.3 présentant les temps des athlètes en finale, il ressort clairement que le temps d'Usain Bolt est une aberration.

Les valeurs aberrantes peuvent être classées en deux sous-catégories que sont les bruits et les anomalies. Les différences entre les deux termes sont finalement une

question d'intérêt humain. Le bruit est une valeur aberrante qui ne nous intéresse pas, tandis que les anomalies sont des valeurs aberrantes qui sont intéressantes, car contenant des informations utiles [39]. Plusieurs raisons peuvent faire apparaître les valeurs aberrantes, cependant les raisons les plus courantes sont des erreurs liées au prétraitement, le bruit, la fraude, les attaques [39]. Un système de détection de fraude a donc pour objectif d'identifier toute valeur aberrante, car susceptible d'être une fraude. Par exemple, si de grosses sommes d'argent sont dépensées l'une après l'autre au cours d'une même journée et qu'il ne s'agit pas de votre comportement habituel, une banque peut bloquer votre carte. Elle constatera un schéma inhabituel dans vos transactions quotidiennes [39]. Cette anomalie peut généralement être liée à une fraude, car les usurpateurs d'identité essaient de voler le plus d'argent possible pendant qu'ils le peuvent. Une fois qu'une anomalie est détectée, elle doit faire l'objet d'une enquête, faute de quoi des problèmes peuvent survenir. Bien qu'existent plusieurs types de valeurs aberrantes, les plus courantes sont les suivants :

- Valeurs aberrantes globales : Lorsqu'un point de données prend une valeur très éloignée de toutes les autres valeurs de points de données dans l'ensemble de données, on peut considérer qu'il s'agit d'une anomalie globale [39]. En d'autres termes, il s'agit d'un événement rare. Par exemple, si vous recevez chaque mois un salaire canadien moyen sur votre compte bancaire, mais qu'un jour vous recevez un million de dollars, l'équipe d'analyse de la banque considérera qu'il s'agit d'une anomalie globale.

- Valeurs aberrantes contextuelles : Une valeur aberrante est qualifiée de contextuelle lorsqu'on ne s'attend pas à observer une telle valeur à un moment précis dans les données [39]. Elles sont généralement temporelles et la même situation observée à différents moments peut ne pas être une valeur aberrante. Par exemple, pour les magasins, il est tout à fait normal d'observer une augmentation de la clientèle pendant la période des fêtes. Toutefois, si une augmentation soudaine se produit en dehors des fêtes, elle peut être considérée comme une valeur aberrante contextuelle.

- Valeurs aberrantes collectives : Elles sont représentées par un sous-ensemble de points de données qui s'écartent du comportement normal [39]. En général, les entreprises technologiques ont tendance à devenir de plus en plus grandes. Certaines entreprises peuvent décliner, mais ce n'est pas une tendance générale. Toutefois, si de nombreuses entreprises affichent simultanément une baisse de leurs revenus au cours de la même période, nous pouvons identifier une valeur aberrante collective.

En outre, une autre difficulté réside dans le fait que les données sont souvent non structurées, ce qui signifie que les informations n'ont pas été organisées d'une manière spécifique pour l'analyse des données [39]. Les documents commerciaux, les courriels ou les images sont des exemples de données non structurées. Dans l'optique de pouvoir détecter les anomalies, nous avons besoin de l'aide de la statistique ainsi que d'outils d'apprentissage automatique. Cela s'explique par l'augmentation de la quantité des données à analyser ainsi que de leur complexité (transactions, textes, images, contenus vidéo, etc.) [39]. Il est tout simplement impossible d'obtenir manuellement des informations significatives à partir d'une telle quantité de données. Les algorithmes d'apprentissage automatique sont donc nécessaires pour la collecte, le nettoyage, la visualisation, l'analyse de ces grands-ensembles de données. Il est donc crucial d'identifier correctement la problématique à résoudre ainsi que le choix minutieux des algorithmes d'apprentissage automatique pouvant nous aider à atteindre cet objectif.

Performances	
Athlète	**Temps (s)**
Usain Bolt	19.30
Shawn Crawford	19.96
Walter Dix	19.98
Brian Dzingai	20.22
Christian Malcolm	20.40
Kim Collins	20.59

TABLE 2.3 – Temps de passage pour la compétition du 200 mètres hommes lors des Jeux olympiques d'été 2008. Source : Wikipedia

2.8 Principale Classification

2.8.1 Apprentissage supervisé

L'apprentissage supervisé consiste à entrainer la machine à l'aide d'ensemble de données étiquetées [63],[10],[61]. Ces informations sont alors utilisées pour prédire les résultats à venir. L'ensemble du processus repose sur la supervision d'où le terme apprentissage supervisé. Chaque entrée est associée à des données étiquetées aidant la machine dans son processus d'apprentissage [63],[66]. Le modèle est alors entrainé en utilisant des données d'entrainement étiquetées puis le modèle est évalué en utilisant des données de test. L'objectif principale de l'apprentissage supervisé est de faire correspondre les variables d'entrées au variables de sortie. L'apprentissage supervisé est largement utilisé dans la détection des fraudes, l'évaluation des risques et le filtrage du spam. l'apprentissage automatique peut être catégorisé en deux types de problèmes :

- Classification : Lorsque la variable de sortie est une réponse binaire et/ou catégorique, des algorithmes de classification sont utilisés pour résoudre les

problèmes [10],[63]. Les réponses peuvent être : Disponible ou Non disponible, Oui ou Non, Rose ou Bleu, etc. Ces catégories sont déjà présentes dans l'ensemble de données et les données sont classées sur la base des ensembles étiquetés fournis pendant la formation du modèle de classification.

- Régression : Contrairement à la classification, un algorithme de régression est utilisé pour résoudre des problèmes où il existe une relation entre les variables d'entrée et de sortie [10],[63]. La régression est utilisée pour faire des prédictions comme par exemples la météo et les conditions du marché.

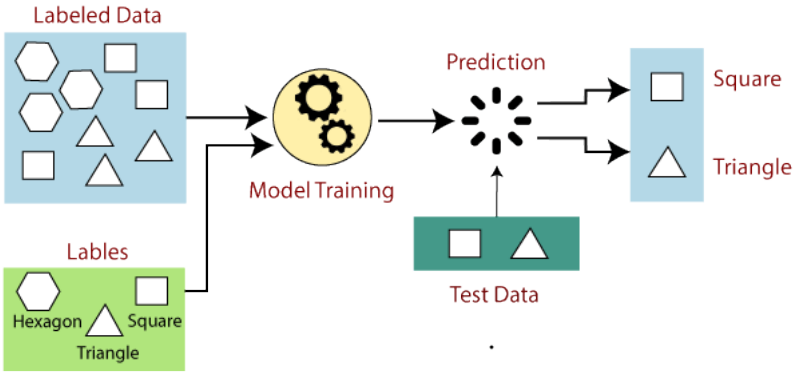


FIGURE 2.3 – Apprentissage supervisé. Source : javatpoint

2.8.2 Apprentissage non supervisé

Contrairement à la technique d'apprentissage supervisé, il n'y a ici aucune supervision. L'entrainement de la machine se fait par l'utilisation des données non étiquetées et non classées, dans ce contexte la machine prédit la sortie sans aucune supervision [10],[63]. Cette méthode est couramment utilisée pour classer ou catégoriser des données non triées en fonction de leurs caractéristiques, similitudes et différences. Cette technique permet de trouver des modèles et des tendances cachés à partir des données d'entrée. L'apprentissage non supervisé peut être classé en deux catégories :

- Le regroupement ou "Clustering" : Il consiste à regrouper des données en se basant sur leurs similarités. Les machines classent les données en fonction de leurs caractéristiques, de leurs similitudes et de leurs différences [10],[63]. Cette technique trouve les groupes inhérents aux données complexes et assure la classification des objets. Elle est couramment utilisée pour comprendre les segments de clientèle et le comportement d'achat, en particulier à travers les régions géographiques.

- L'association : Dans cette méthode les machines trouvent des relations et des connexions intéressantes entre les variables au sein de grands ensembles de données qui sont fournis en entrée [10],[63]. Par exemples comment une donnée dépend-elle d'une autre ?, quelle est la procédure pour mettre en correspondance les variables ? , comment ces connexions peuvent-elles aboutir à un bénéfice ?, ce sont les principales considérations de cette technique d'apprentissage. Cet algorithme est particulièrement populaire dans l'exploration de l'utilisation du Web et la vérification du plagiat dans les travaux de doctorat.

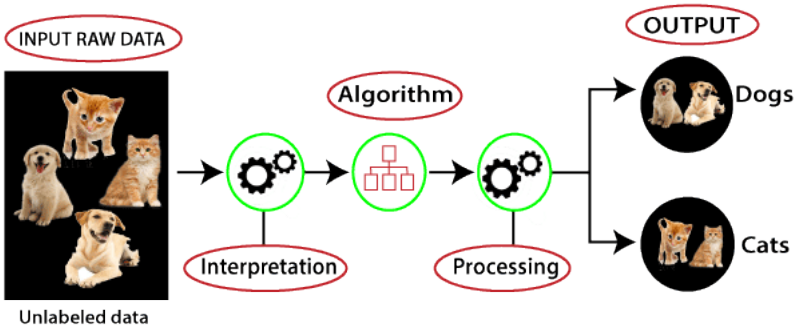


FIGURE 2.4 – Apprentissage non supervisé. Source : javatpoint

2.8.3 Apprentissage semis-supervisé

Cette technique a été créée en tenant compte des avantages et des inconvénients des méthodes d'apprentissage supervisé et non supervisé. Pendant la période d'entrainement du modèle, une combinaison d'ensembles de données étiquetées et non étiquetées est utilisée pour entrainer les machines [63],[39]. Cependant, dans le monde réel, la plupart des ensembles de données d'entrée sont des données non étiquetées. L'avantage de cette méthode est qu'elle utilise toutes les données disponibles, et pas seulement les informations étiquetées. Tout d'abord, les données similaires sont regroupées. Cette opération est réalisée à l'aide d'un algorithme d'apprentissage non supervisé. Cela permet d'étiqueter toutes les données non étiquetées. On distingue deux catégories d'apprentissage semis-supervisé :

- L'apprentissage autosupervisé : Dans cette technique l'orsque le modèle reçois des données non structurées, il génère de façons automatique des étiquetés qui sont par la suite utilisés pour l'entrainement du modèle dans les prochaines itérations [39]. Cette opération résout les problèmes posés par la dépendance excessive des données étiquetées.

- C'est un type d'apprentissage supervisé dans lequel un modèle reçoit un ensemble de sacs étiquetés individuellement [82]. Dans ce contexte, un sac porte l'étiquette

négative si toutes les instances contenues dans ce sac sont négatives. En revanche, un sac est étiqueté comme positif si au moins une instance qu'il contient est positive. Dans ce contexte, le modèle, grâce à une collection de sacs étiquetés, tente soit d'induire un concept qui étiquettera correctement les instances individuelles, soit d'apprendre à étiqueter les sacs sans induire le concept.

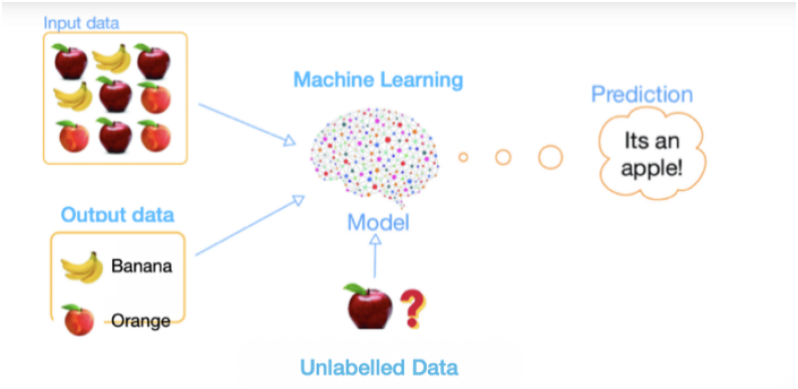


FIGURE 2.5 – Apprentissage semis-supervisé. Source : neptune.ai

2.8.4 Apprentissage par renforcement

Dans l'apprentissage par renforcement il n'y a pas la notion de données étiquetées, dans ce type d'apprentissage, la machine apprend seulement à partir de l'expérience [63]. C'est une méthode d'apprentissage automatique basée sur la récompense des comportements souhaités ou la punition des comportements non souhaités. En général, un agent d'apprentissage par renforcement est capable de percevoir et d'interpréter son environnement, de prendre des mesures et d'apprendre par essais et erreurs comme l'illustre la figure 2.6.

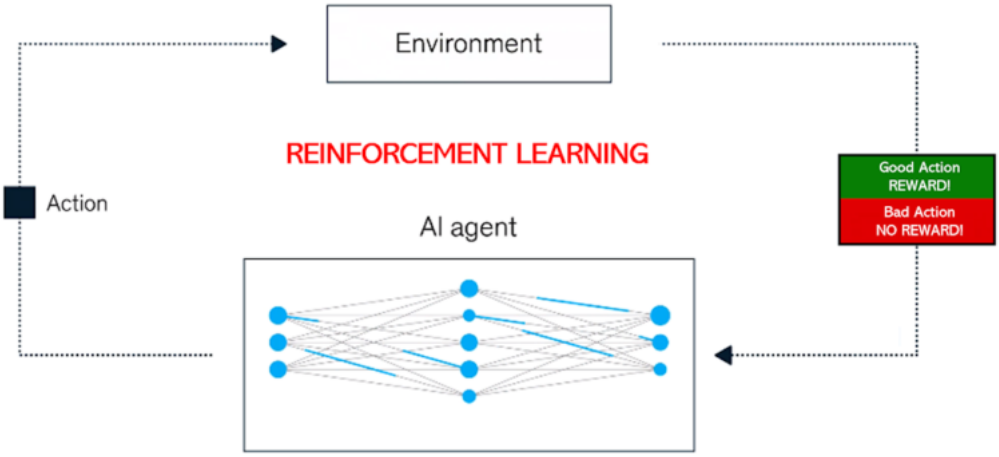


FIGURE 2.6 – Apprentissage par renforcement. Source : Enterrasolutions

2.9 Méthodes basées sur l'apprentissage automatique

2.9.1 Naïve Bayes

Naive Bayes est un modèle de probabilités conditionnel reposant sur le théorème de Bayes avec de fortes hypothèses d'indépendance (naive) entre chaque paire de caractéristiques. Il assigne des probabilités $p(C_k|x_1, ..., x_n)$ pour chacun des K résultats ou classes C_k étant donné une instance de problème à classer, représentée par un vecteur $X = (x_1, ..., x_n)$ encodant n caractéristiques (variables indépendantes) [5]. Naive Bayes est basé le théorème suivant $p(C_k|X) = \frac{p(C_k)p(X|C_k)}{p(X)}$. Dans cette fraction, nous allons-nous intéresser uniquement aux numérateurs, car le dénominateur ne dépend pas de C et la valeur de chaque caractéristique x_i est donnée ce qui veut dire que le dénominateur est effectivement une constante. Le numérateur est équivalent au modèle de probabilité conjointe $p(C_k, x_1, ..., x_n)$.

Il est couramment utilisé dans de nombreuses applications telles que la classification des documents ou textes, le filtrage de spam,etc. Considérons l'exemple suivant, un fruit peut être considéré comme une pomme s'il est rouge, rond et d'un diamètre d'environ 10 cm. Un classificateur Naive Bayes considère que chacune des caractéristiques contribue de façon indépendante a la probabilité que ce fruit soit une pomme sans tenir compte des corrélations éventuelles entre les caractéristiques (couleur, rondeur et diamètre). L'un des avantages de l'algorithme Naive Bayes est qu'il nécessite une petite quantité de données d'entraînement pour pouvoir estimer les paramètres nécessaires à la classification, cependant, ses performances peuvent être affectées en raison de ses fortes hypothèses sur l'indépendance des caractéristiques.

2.9.2 Régression logistique

La régression logistique est un modèle d'apprentissage supervisé. Elle est également un modèle statistique utilisé pour résoudre les problèmes de classification dans l'apprentissage automatique. Elle utilise généralement une fonction logistique pour estimer les probabilités, également appelée fonction sigmoïde définie mathématiquement par l'équation $\frac{1}{1+exp^{-z}}$ [63],[5].On peut voire cela sur la figure 2.7.

L'hypothèse de linéarité entre les variables dépendantes et indépendantes est considérée comme un inconvénient majeur de la régression logistique. Elle peut être utilisée à la fois pour les problèmes de classification et de régression, mais elle est plus couramment utilisée pour la classification.

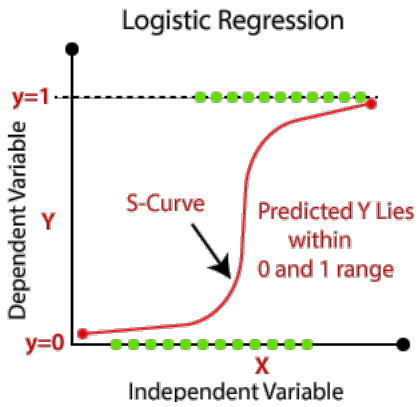


FIGURE 2.7 – Processus de régression logistique. Source : analyticsvidhya

2.9.3 K-plus proches voisins

L'algorithme des K-plus proches voisins est un algorithme supervisé qui est utilisé pour résoudre des problèmes de classification et de régression [10]. C'est un algorithme non paramétrique parce qu'il ne fait aucune hypothèse sur la distribution sous-jacente des données. En d'autres termes, le K-plus proches voisins tente de déterminer le groupe auquel appartient un point de données en examinant les points de données qui l'entourent [63]. Il est également considéré comme un algorithme paresseux (lazy learning) car il n'effectue aucun apprentissage lorsque vous fournissez les données d'apprentissage [63].

Considérons un exemple dans lequel nous avons deux groupes de points A et B l'algorithme examine l'état des points de données situés à proximité. Si la majorité des points de données sont dans le groupe A, il est très probable que le point de données en question soit dans le groupe A et vice-versa. On peut considérer le fonctionnement de l'algorithme comme consistant a classé un point de données en examinant le point de données le plus proches d'où son nom du K-plus proches voisins. Voici la figure 2.8 illustrant le fonctionnement de l'algorithme K-plus proches voisins.

La distance est utilisée par l'algorithme K-plus proches voisins (KNN) pour déterminer la similarité entre deux vecteurs. Le calcul de la distance permet de mesurer le degré de différence entre deux vecteurs. Les distances les plus couramment utilisées sont les suivantes [41] [33].

- Distance euclidienne : C'est la mesure de distance la plus couramment utilisée symbolisée par la formule $d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$ ou x et y sont des vecteurs.

- Distance de Manhattan : C'est également une distance populaire mesurant la valeur absolue entre deux points, elle est représentée par la formule $d(x, y) = (\sum_{i=1}^{n} |x_i - y_i|)$ ou x et y sont des vecteurs.

- Distance de Minkowski : C'est la forme généralisée des mesures de distance Euclidienne et de Manhattan. Elle est représentée par la formule $d(x, y) = (\sum_{i=1}^{n} |x_i - y_i|)(1/p)$ ou x et y sont des vecteurs, p est un paramètre permettant la création d'autres mesures de distance, quand p est égal à deux c'est la distance euclidienne et quand p est égal à un c'est la distance de Manhattan.

- Distance de Hamming : Cette technique est généralement utilisée avec des vecteurs booléens ou des chaînes de caractères et permet d'identifier les points où les vecteurs ne correspondent pas. Elle est représentée par la formule $D_H = (\sum_{i=1}^{k} |x_i - y_i|)$ ou x et y sont des vecteurs, X=Y on a D=0, $X \neq Y$ on a $D \neq 1$.

L'algorithme KNN fonctionne sur le principe de la similarité. Il prédit l'étiquette ou la valeur d'un nouveau point de données en tenant compte des étiquettes ou des valeurs de ses K voisins les plus proches dans l'ensemble de données d'apprentissage. Les principales étapes de son fonctionnement sont les suivantes [33] :

- La sélection de la valeur optimale de K, ou K représente le nombre de voisins les plus proches à prendre en compte lors de la prédiction.

- Calcul de la distance pour mesurer la similarité entre les points de données cibles et les points de données d'entraînement.

- Trouver les plus proches voisins , les k points de données présentant les distances les plus faibles par rapport au point cible sont les plus proches voisins.

- Voter pour la classification(réalisation d'un vote à la majorité) ou prendre la moyenne pour la régression.

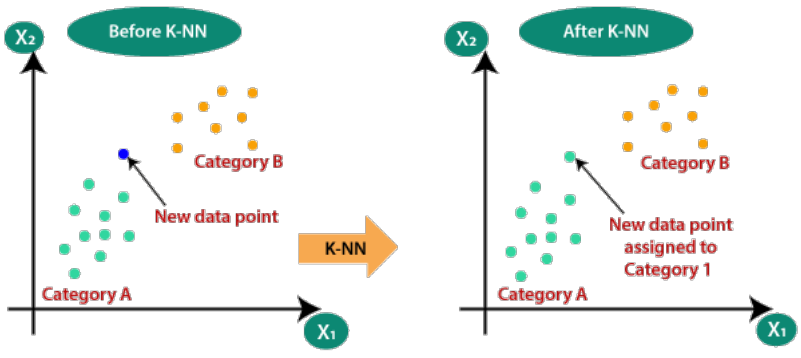


FIGURE 2.8 – Fonctionnement de l'algorithme K-plus proches voisins. Source : javatpoint

2.9.4 Machine à vecteur de support(SVM)

L'algorithme machine à vecteur de support plus connu sous le terme anglais "SVM" est un algorithme d'apprentissage supervisé qui recherche un hyperplan à marge maximale permettant de classer les échantillons d'entrée en deux classes comme l'illustre la figure 2.9 [10]. Il peut contribuer à résoudre les problèmes linéaires et non linéaires et fonctionne bien pour de nombreux problèmes pratiques [63].

Le principe du SVM est très simple, l'algorithme crée une ligne ou un hyperplan qui sépare les données en classes si cela est possible. Considérons la figure 2.9 qui présente un ensemble de données contenant des rectangles rouges et des cercles bleus. Notre objectif est de pouvoir faire la classification de ces deux catégories d'objet , pour cela nous devons trouver la ligne idéale qui sépare les données en deux classes distinctes. Il est important de noter que dans notre exemple, il existe une infinité de lignes pouvant séparer les données en deux classes, SVM a pour but de trouver la ligne optimale pouvant séparer nos données en deux classes.

En analysant la figure 2.10 présentant deux hyperplans pouvant séparer nos données en deux classes, la question qui nous vient à l'esprit est la suivante laquelle de ses deux lignes représente le mieux la séparation des données si votre choix, c'est porter sur la ligne jaune qui est un séparateur plus général contrairement a la ligne verte qui est plus proche des données de la classe rouge. Pour déterminer, la ligne idéale pour la séparation de nos données, SVM utilise la méthodologie qui est la suivante :

- Trouver les points les plus proches de la ligne pour les deux classes. Ces points sont appelés des vecteurs de support qui nous seront très utiles dans la suite de la méthodologie.

- Calculer la distance entre la ligne et les vecteurs de support. Le but de l'algorithme SVM est de maximiser cette distance. L'hyperplan pour lequel la marge est maximale est l'hyperplan optimal, on peut voir cela sur la figure 2.11.

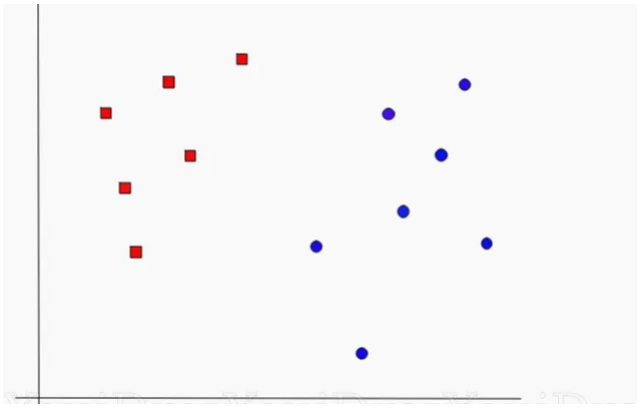


FIGURE 2.9 – trouver l'hyperplan qui sépare cet ensemble de données en catégories rouge et bleue

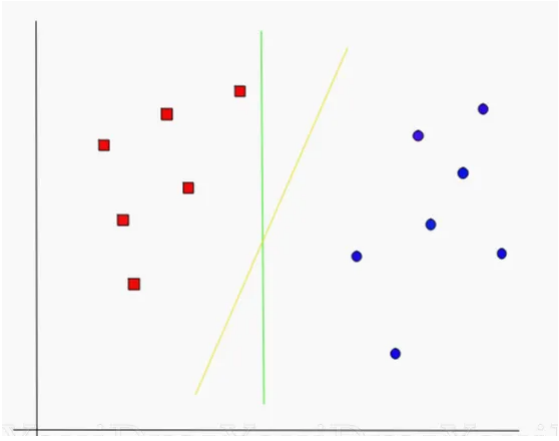


FIGURE 2.10 – Hyperplans présentant la séparation des données. Source : towardsdatascience

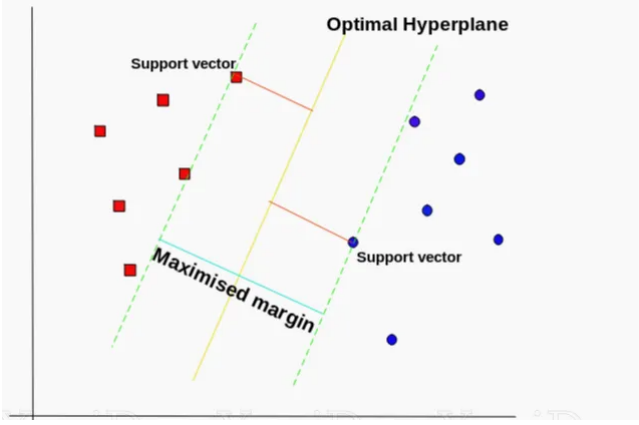


FIGURE 2.11 – SVM méthodologie pour trouver la ligne optimale. Source : towardsdatascience

2.9.5 Arbre de décision, cas particulier du Random Forest

L'arbre de décision est une méthode d'apprentissage supervisée qui est utilisée pour les tâches de classification et de régression. Cet algorithme fonctionne en triant l'arbre

de la racine à certain nœud de la feuille, comme le montre la figure 2.12 [63]. Il est important de noter qu'il existe plusieurs variations de l'algorithme d'arbre décisionnel cependant dans notre contexte, nous allons nous intéresser au cas du Random Forest.

Random Forest est une méthode d'ensemble basée sur un ensemble d'arbres de décision pouvant être utilisés pour résoudre à la fois des problèmes de classification et de régression [63],[5]. Elle utilise la méthode d'ensemble plus connu sur le terme "bagging" consistant à entrainer de façons indépendantes chaque classificateur en considérant un sous-ensemble (X_i, Y_i) de l'ensemble de données originales(X, Y). Cet algorithme utilise l'assemblage en parallèle qui consiste à l'utilisation de plusieurs arbres de décision, comme le montre la figure 2.13 [63]. Cela consiste à choisir un sous-ensemble aléatoire de caractéristiques pour chaque arbre dans le but de trouver le seuil qui sépare le mieux des données. En appliquant cette méthode, nous allons obtenir plusieurs arbres qui seront entraînés de façon indépendante et chaque arbre produira une prédiction différente. Il existe plusieurs interprétations des résultats qui seront obtenus :

- L'approche la plus courante est basée sur un vote majoritaire consistant à choisir la classe ayant eu le plus de vote comme la classe correcte [63].

- Une autre alternative consiste à choisir la moyenne des résultats comme étant la classe correcte [63].

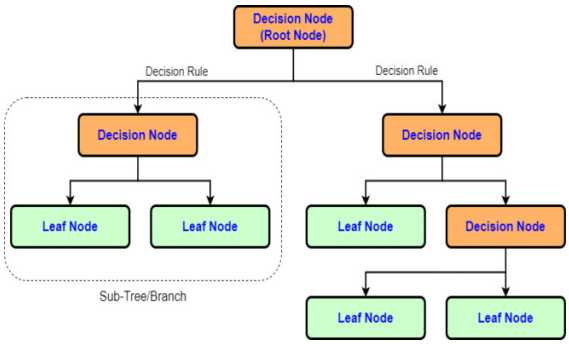


FIGURE 2.12 – Arbre de décision. Source : researchgate

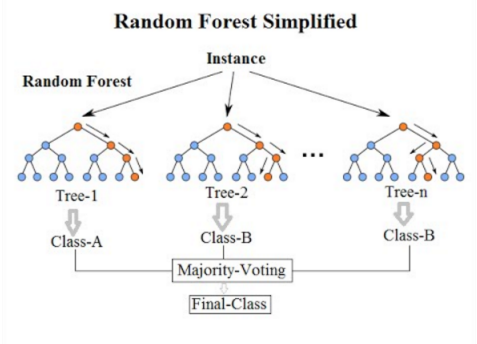


FIGURE 2.13 – Diagramme de décision de l'algorithme Random Forest. Source : wikipedia

2.10 Méthodes basées sur l'apprentissage profond et méthodes d'ensembles

2.10.1 Autoencodeur

L'autoencoder est un algorithme de réseaux de neurones non supervisés, qui est composé d'une couche latente qui représente une version comprimée des données d'entrée originales [27],[79]. Ils sont largement utilisés pour la réduction de dimension, la compression d'image , le débrutage d'image et l'extraction de caractéristiques. Ils sont également utilisés pour la détection d'anomalies, domaine dans le quelle l'autoencoder a produit de bons résultats pour sa fiabilité [67],[65],[79].

Les autoencodeurs sont également des modèles génératifs qui sont capables de générer aléatoirement de nouvelles données similaires aux données d'entrainement. La figure 2.14 nous montre l'architecture de base de l'autoencodeur qui est constitué deux grandes parties qui sont les suivants :

- L'encodeur compresse les données d'un espace de dimension supérieure vers un espace de dimension inférieure également appelé espace latent. En analysant la figure 2.14 l'encodeur prend en paramètre un vecteur d'image X et le transmet par la suite à l'espace latent sous la forme d'un vecteur compressé Z, puis le décodeur tente alors de reconstruire l'image d'entrée à partir de la représentation comprimée [27],[79].

- Le décodeur permet de s'assurer que l'espace latent peut capturer la plupart des informations de l'espace de l'ensemble des données, en le forçant à produire ce qui a été donné en entrée au décodeur. Le décodeur permet en résumés de reconstruire les données d'entrées [27],[79].

De façons mathématiques, un autoencodeur peut être défini comme un ensemble de deux espaces, l'espace des messages décodés x, l'espace des messages encodés Z , en général X' et Z deux espaces euclidiens. La function de l'encodeur $E_\emptyset : X \to Z$ paramétré par $\emptyset$, la function du décodeur $D_\theta : Z \to X'$ paramétré par θ. Pour tout x $\in$ X, on a z = $E_\emptyset(x)$ et cela correspond a l'espace latent, réciproquement pour tout z $\in$ Z on a x'= $D_\theta(z)$ ce qui correspond aux messages décodés. L'entrainement d'un autoencodeur peut se résumer en deux fonctions, une fonction définie par une distribution de probabilité de référence U sur X et une fonction de la qualité de la reconstruction $X.X \to [0; \infty]$ telle que d(x,x') mesure la différence entre x et x'.

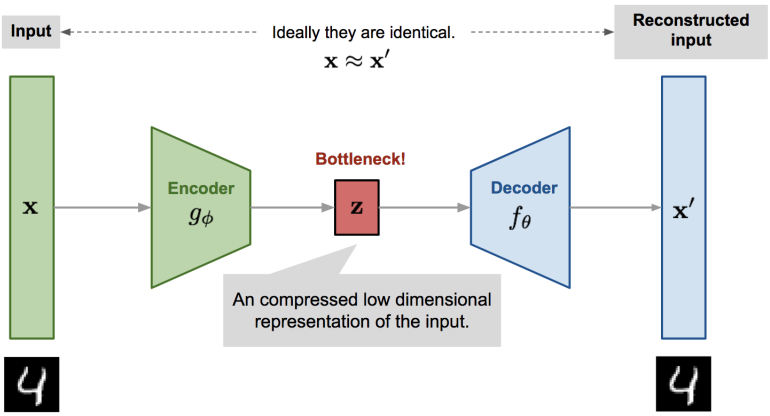


FIGURE 2.14 – Illustration de l'architecture de l'Autoencoder. Source : blog de Lilianweng

2.11 XGBoost

XGBoost est un algorithme de classification et également une méthode d'apprentissage ensembliste [63]. L'apprentissage d'ensemble offre une solution systématique pour combiner le pouvoir prédictif de plusieurs modèles [63]. Le résultat est un modèle unique qui donne les résultats agrégés de plusieurs modèles. Les modèles qui forment l'ensemble, également connus sous le nom d'apprenant de base, peuvent provenir du même algorithme d'apprentissage ou d'algorithmes d'apprentissage différents. Bien qu'il existe plusieurs méthodes d'apprentissage par ensemble, les plus courantes sont le Bagging et boosting, ces deux techniques peuvent être utilisées avec plusieurs modèles statistiques cependant, c'est avec les arbres de décision qu'elles sont le plus utilisé [63],[5].

Avant de chercher à comprendre le fonctionnement de l'algorithme XGBoost, il est nécessaire de comprendre les concepts suivants :

- Bien que les arbres de décision soient l'un des modèles les plus faciles à interpréter, leur comportement est très variable. Considérons un seul ensemble de données d'apprentissage que nous divisons aléatoirement en deux parties. Utilisons maintenant chaque partie pour former un arbre de décision afin d'obtenir deux modèles. L'ajustement de ces deux modèles donnerait des résultats différents. En raison de ce comportement, les arbres de décision sont associés à une variance élevée. L'agrégation de type "bagging" ou "boosting" permet de réduire la variance de n'importe quel apprenant. Plusieurs arbres de décision générés en parallèle constituent les apprenants de base de la technique de bagging. Les données échantillonnées avec remplacement sont transmises à ces apprenants pour la formation. La prédiction finale est la moyenne des résultats de tous les apprenants.

- Dans le cas du "Boosting", les arbres sont construits de manière séquentielle, de sorte que chaque arbre suivant vise à réduire les erreurs de l'arbre précédent [23],[63]. Chaque arbre apprend de ses prédécesseurs et met à jour les erreurs résiduelles. Ainsi, l'arbre suivant dans la séquence apprendra à partir d'une version actualisée. Les apprenants de base de la technique de "Boosting" sont des apprenants faibles dont le biais est élevé et dont le pouvoir prédictif est à peine supérieur à celui d'une supposition aléatoire. Chacun de ces apprenants faibles apporte des informations essentielles à la prédiction, ce qui permet à la technique de renforcement de produire un apprenant fort en combinant efficacement ces apprenants faibles. L'apprenant fort final réduit à la fois le biais et la variance. Cette technique a également comme avantage de résoudre le problème de surapprentissage(overfitting) que présente la technique de "Bagging".

2.11.1 Algorithme de renforcement du gradient

L'Algorithme de renforcement du gradient, plus connu sur le terme anglais de "LightGBM ", est une méthode d'ensemble de renforcement du gradient basée sur les arbres de décision [28]. À l'image des autres algorithmes d'arbres de décision, il peut être utilisé à la fois pour résoudre des problèmes de classification et de régression. L'une des particularités du LightGBM est sa performance élevée avec les systèmes distribués, en plus de cela il possède plusieurs fonctionnalités comme l'entrainement en parallèle, la régularisation, le Bagging [28]. La particularité de cet algorithme repose sur les fonctionnalités suivantes :

- Échantillonnage unilatéral basé sur le gradient : C'est une technique utilisée dans LightGBM pour sélectionner un sous-ensemble de données à utiliser lors de l'apprentissage d'un modèle de renforcement du gradient.

- Offre groupée de fonctionnalités exclusives : L'algorithme combine des caractéristiques exclusives pour réduire la dimensionnalité, ce qui le rend plus rapide et plus efficace.

2.12 Techniques de génération de données synthétiques

2.12.1 SMOTE

SMOTE est une technique statistique de suréchantillonnage des minorités synthétiques permettant la génération de nouvelles données synthétiques des classes minoritaires [22],[12].Elle est apparue pour la première fois dans l'article de recherche

intitulé "SMOTE : Synthetic Minority Over-sampling Technique" paru en 2022, suite à cela elle a connu de nombreuses évolutions conduisant à l'apparition de plusieurs variantes de SMOTE plus performantes [22]. Les principales étapes de la génération de nouvelles données synthétiques sont les suivantes :

1. Sélectionner un échantillon de classe minoritaire dans l'ensemble de données original.

2. Trouver les k voisins les plus proches de la classe minoritaire dans l'espace des caractéristiques.

3. Sélectionner aléatoirement l'un des k voisins les plus proches.

4. Générer un nouvel échantillon synthétique en interpolant entre l'échantillon de la classe minoritaire sélectionnée et le voisin choisi au hasard.

5. Répéter les étapes 1 à 4 jusqu'à ce que le nombre désiré d'échantillons synthétiques soit généré.

2.12.2 Réseaux adversariaux génératifs

Les réseaux de neurones GAN sont une famille de framework d'apprentissage automatique conçue initialement par Ian Goodfellow et ses collègues en juin 2014, dans lequel deux réseaux de neurones adverses s'affrontent sous la forme d'un jeu à somme nulle, où le gain d'un agent est la perte d'un autre agent [35]. Considérons un ensemble de données d'entraînement, avec cette technique, nous pouvons générer de nouvelles données avec les mêmes statistiques que les données d'entraînement. Prenons l'exemple d'un modèle GAN entraîné sur des données des visages humains, ce modèle pourra à la suite de son entraînement générer de nouveaux visages d'humain.

Les réseaux GAN sont constitués de deux sous modèles qui sont le générateur et le discriminateur qui fonction selon le principe suivant, le générateur génère de faux échantillons à l'aide d'un bruit aléatoire et quant aux discriminateurs son rôle est de faire la différence entre les vraies et les faux échantillons en fonction des informations renvoyées par le discriminateur, le générateur améliore les faux échantillons générer de telle sorte qu'il devient difficile d'identifier les vraies des faux échantillons. Bien qu'il existe plusieurs variations du GAN dans notre mémoire nous ne nous intéresserons qu'aux variations les plus connues [35],[53],[80],[54].

Vanilla GAN

Dans ce type de réseau de neurones adverses, le modèle de réseau génératif est opposé à un adversaire qui est un modèle discriminant qui apprend à déterminer si un échantillon provient des données ou du modèle génératif [35]. Le modèle génératif peut être comparé à une équipe de faux-monnayeurs, qui tente de produire de la fausse monnaie et de l'utiliser sans être détectée, tandis que le modèle discriminant est comparé à la police, qui tente de détecter la fausse monnaie. La compétition dans ce jeu pousse les deux adversaires à améliorer leurs méthodes jusqu'à ce que les contrefaçons soient indiscernables des articles authentiques.

Le modèle génératif génère les échantillons en faisant passer un bruit aléatoire à travers un réseau de neurones multicouche également dans cette architecture, le modèle discriminant est un réseau de neurones multicouche [35].Dans ce cas, nous pouvons former les deux modèles en utilisant uniquement les algorithmes de rétropropagation et d'abandon, qui ont fait leurs preuves, et échantillonner à partir du modèle génératif en utilisant uniquement la propagation vers l'avant. L'architecture de réseaux de neurones adverses est facilement applicable lorsque les deux modèles sont des réseaux de neurones de multicouche. Pour connaitre la distribution p_g du générateur sur les données x, nous devons d'abord définir les variables d'entrées de bruit $p_z(z)$ puis faire une correspondance avec l'espace de données $G(z;\theta_g)$ où G est une fonction différentiable représentée par un réseau de neurones multicouche avec un paramètre θ [35]. Nous définissons également un autre réseau de neurones multicouche $D(x;\theta_d)$ qui produit un seul scalaire, D_x représente la probabilité que x vienne des données plutôt que de p_g [35]. Nous entrainons le discriminateur pour maximiser la probabilité d'attribuer la bonne étiquette aux exemples d'apprentissage et aux échantillons du générateur. Nous entrainons de façons simultanées les générateurs pour minimiser le log(1-D(G(z))).

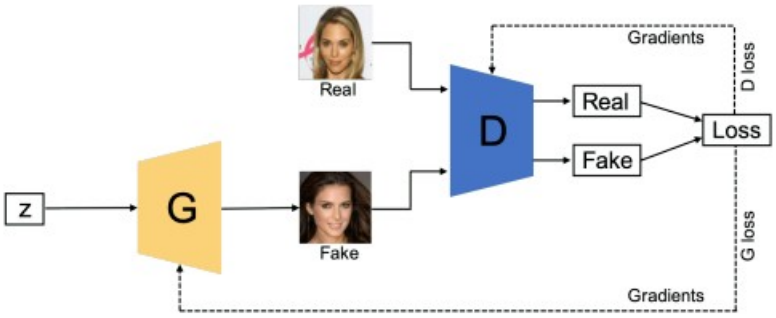


FIGURE 2.15 – Architecture du Vanila GAN. Source : opengenus

GAN conditionnel

Les réseaux de neurones adverses génératifs (GAN) peuvent être étendus a un modèle conditionnel si on ajoute à la fois aux générateurs et aux discriminateurs des

informations supplémentaires y [54]. y peut être toute information auxiliaire telle que les étiquetés de la classe ou des données modales. La conditionnalité est faite en passant aux générateurs et aux discriminateurs une couche supplémentaire d'entrée. Le bruit initiale $p_z(z)$ du générateur et y sont combinés dans une représentation cachée commune, pour le discriminateur x et y sont présent en tant qu'entrée ou x représente les caractéristiques [54].

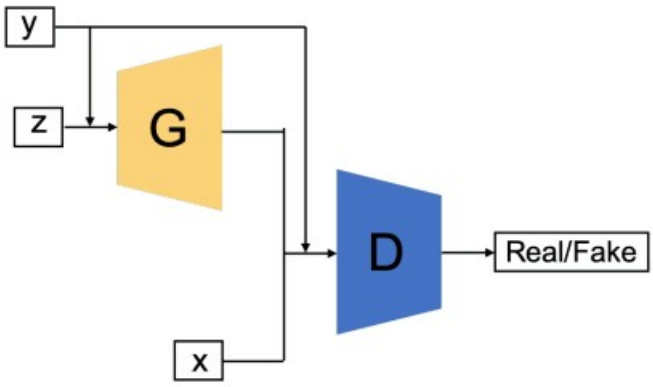


FIGURE 2.16 – Architecture du GAN conditionnel. Source : opengenus

Wasserstein GAN

Le Wasserstein GAN où WGAN fut introduit pour la première fois par Martin Arjovsky en 2017 dans son article, dans lequel il présente une meilleure méthode pour entraîner le générateur pour mieux approcher de la distribution des données observées dans un ensemble de données [53]. Le WGAN repose essentiellement sur un fondement mathématique connu sous le nom de distance de Wasserstein principe selon lequel le discriminateur joue le rôle de critiques qui fournit un retour d'information sur la distance entre les données générées et les données réelles permettant alors un meilleur ajustement du générateur [85]. Le GAN de Wasserstein est obtenu en utilisant la métrique de Wasserstein, qui répond à un « théorème de représentation duale » connu sous le nom de théorème de Kantorovich-Rubenstein. Lorsque l'espace de probabilité Ω est un espace de métrique alors pour toute valeur fixe de k>0,$W_1(u,v) = \frac{1}{K_{|f|_L \leq K}} sup E_{x\ \mu}[f(x)] - E_{y\ \nu}[f(y)]$ ou $|*|L$ est la norme de Lipschitz. Grâce à la dualité Kantorovich-Rubenstein, la définition du GAN de Wasserstein est claire [85] :

- Le Wasserstein GAN est défini par un espace de probabilité (Ω, B, ν_{ref}) ou Ω est un espace de métrique et une constante K>0.
- On a deux joueurs qui sont le générateur et le discriminateur qu'on appelle critique.
- L'ensemble des stratégies du générateur est l'ensemble de toutes les mesures de probabilité μ_G sur (Ω, B).

- L'ensemble stratégique du discriminateur est l'ensemble des fonctions mesurables de type $D : \Omega \to \mathbb{R}$ avec une norme de Lipschitz bornée : $|D|_L \leq K$.
- Le jeu de du GAN de Wasserstein est un jeu à somme nulle, avec une fonction objective $L_{WGAN_{\mu_G,D}} = E_{x\ \mu_G}[D(x)] - E_{y\ \nu_{Ref}}[D(x)]$.
- Le générateur passe en premier et le discriminateur en second. Le générateur vise à minimiser l'objectif, et le discriminateur vise à maximiser l'objectif : $minmax L_{WGAN}(\mu_G, D)$.
- Pour toute stratégie du générateur Ug , la réponse optimale du discriminateur est D^* tel que $L_{WGAN}(\mu_G, D^*) = K.W_1(\mu_G, \mu_{ref})$.
- La stratégie optimale du générateur est de minimiser $W_1(\mu_G, \mu_{ref})$,avec comme finalités $\mu_G = \mu_{ref}$.

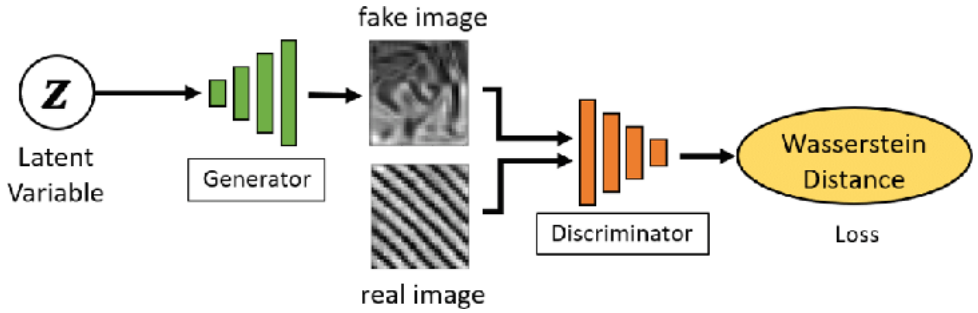


FIGURE 2.17 – Architecture du Wasserstein GAN. Source : researchgate

2.13 Métriques de performance des algorithmes d'apprentissage automatique

2.13.1 Types et partage des données

Dans l'optique d'avoir une évaluation juste d'un modèle, il est recommandé d'entraîner le modèle sur un ensemble de données d'entraînement et d'effectuer les tests sur un jeu de données indépendant et non-vue par le modèle que l'on appelle données de test [1]. En procédant de cette manière, cela nous permet de connaître l'efficacité réelle de notre modèle en le testant sur des données qu'il n'a jamais vues qui sont plus proches des données que le modèle rencontrera dans le monde réel. Nous pouvons rencontrer trois types d'ensembles de données dans l'apprentissage automatique :

- Ensemble d'entraînement : Ensemble de données sur lequel le modèle est entraîné.

- Ensemble de validation : Ensemble de données utilisé pour ajuster les hyperparamètres du modèle. Un ensemble de validation est souvent appelé ensemble de développement.

- Ensemble d'évaluation ou ensemble de test : Ensemble de données utilisé pour évaluer les performances du modèle.

Pour illustrer ce concept, considérons un jeu de donnée "A" que l'on divise en trois parties, dont 70% des données, sont alloués à l'entraînement du modèle, 20% sont alloués comme données de validation du modèle, 10% sont alloués comme données de test. Il n'existe pas un ratio spécifique pour le partage des données, cependant il est important de nous assurer que l'ensemble de tests est suffisamment grand pour fournir des résultats d'évaluation statistiquement significatifs.

2.13.2 Validation croisée

La validation croisée (Cross-validation) est une technique utilisée pour estimer la précision d'un modèle dans un scénario réel [1],[43]. Il est généralement utilisé pour détecter l'ajustement excessif, c'est-à-dire l'incapacité d'un modèle à généraliser des données, en particulier lorsque la quantité de données est limitée. On distingue principalement deux types de validation croisés qui sont les suivantes [1],[43] :

- Holdout : Dans cette méthode, les points de données sont répartis de manière aléatoire entre deux ensembles, généralement appelés ensemble d'apprentissage et ensemble de test, respectivement. Nous entraînons le modèle sur l'ensemble d'entraînement et nous testons le modèle sur l'ensemble de test.

- k-fold : Cela fonctionne de la manière suivante :

  -Nous mélangeons les données de manière aléatoire.

  -Nous divisons toutes les données en k parties, également appelées "folds". Nous formons le modèle sur k-1 "folds" et l'évaluons sur le "folds" restant. Nous enregistrons les performances de ce modèle à l'aide de la métrique d'évaluation du modèle que nous avons choisie, puis nous rejetons ce modèle.

  -Nous répétons ce processus k fois, en proposant à chaque fois un sous-ensemble différent à tester. Nous calculons la moyenne des valeurs de la métrique d'évaluation (par exemple, la précision) de tous les modèles précédents. Cette moyenne représente la mesure de performance globale du modèle.

La validation croisée k-fold est principalement utilisée lorsque nous disposons d'un nombre limité de points de données, par exemple 100 points. L'utilisation de 5 ou 10 "folds" est la plus courante lors de la validation croisée.

2.13.3 Métriques de performance

L'évaluation des performances des modèles d'apprentissage automatique est une tâche très importante, qui nous permet d'identifier les problèmes critiques que les modèles peuvent cacher. Cette affirmation est encore plus véridique dans le contexte de la détection des fraudes, car les données de transactions financières sont très déséquilibrées. Ce déséquilibre des données de transactions financières conduit les algorithmes d'apprentissage automatique à ne prédire que les données de la classe majoritaire (transactions normales) alors que la cible à prédire est la classe minoritaire (transactions frauduleuses) [1],[43]. Ce dysfonctionnement donne l'illusion que les algorithmes sont très performants alors que ce n'est pas le cas, il est donc crucial pour nous d'avoir des métriques qui permettent d'évaluer les performances réelles de nos modèles [1],[43],[56]. Les métriques de performances auxquelles nous nous intéresserons sont les suivantes :

- Les vrais positifs (TP) représentent les résultats pour lesquels notre modèle prédit correctement la classe positive. Dans le contexte de la détection des fraudes, les vrais positifs représentent les transactions frauduleuses qui ont été classées comme étant des transactions frauduleuses.

- Les faux positifs représentent les résultats pour lesquels le modèle prédit incorrectement la classe positive. Dans le contexte de la détection des fraudes, les faux positifs (FP) indiquent le nombre de transactions légitimes classées à tort comme frauduleuses.

- Les vrais négatifs représentent les résultats pour lesquels le modèle prédit correctement la classe négative. Dans le contexte de la détection des fraudes, les vrais négatifs (TN) représentent les transactions légitimes classées comme légitimes.

- Les faux négatifs (FN) représentent les résultats pour lesquels le modèle prédit incorrectement la classe négative. Dans le contexte de la détection des fraudes, les faux négatifs (FN) indiquent les transactions frauduleuses mal classées comme légitimes.

- L'accuracy est la fraction de prédictions correctes sur l'ensemble des prédictions faites par le modèle. L'expression mathématique de l'accuracy est : $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$. L'accuracy n'est pas une bonne mesure à prendre en compte lorsque nos données sont déséquilibrées, car elle ne fait pas la distinction entre le nombre d'exemples correctement classés de différentes classes. Si une classe est beaucoup plus fréquente que l'autre, l'accuracy peut être élevée même

si le modèle n'est pas très bon pour prédire la classe rare, c'est-à-dire, dans notre contexte, les transactions frauduleuses.

- La précision est un indicateur de la performance d'un modèle. Elle indique la fréquence à laquelle notre modèle prédit correctement la classe cible. L'expression mathématique de la précision est la suivante : $Précision = \frac{TP}{TP+TN+FP+FN}$.

- Le rappel est une mesure qui évalue la fréquence à laquelle un modèle d'apprentissage automatique identifie correctement des instances positives parmi tous les échantillons positifs réels de l'ensemble de données. L'expression mathématique de la précision est : $Rappel = \frac{TP}{TP+TN}$.

- Le ScoreF1 est une mesure de la moyenne harmonique de la précision et du rappel, qui permet de mieux comprendre les performances du modèle. L'expression mathématique du scoreF1 est $scoreF1 = \frac{Precision \times Recall}{Precision+Recall}$.

- La métrique F-beta est une métrique robuste pour les données équilibrées et déséquilibrées, c'est la forme généralisée du score F1. Plus simplement, il s'agit d'une moyenne harmonique pondérée de la précision et du rappel, où le paramètre bêta contrôle l'importance relative de la précision et du rappel. L'expression mathématique de la métrique F-beta est : $F - beta = \frac{(1+\beta^2)\times(Precision \times Recall)}{(\beta^2 \times Precision)+Recall}$.

- Le score de précision équilibrée peut être défini comme le rappel moyen obtenu dans chaque classe. Cette mesure peut être utilisée pour traiter des ensembles de données déséquilibrés. Considérons un problème de classification binaire dans lequel nous avons deux classes A et B. Dans ce contexte, la formule pour le score de précision équilibrée est la suivante : $BalancedAccuracy = (RecallA + RecallB)/2$.

2.14 Apprentissage automatique contradictoire

Avec l'avènement de l'intelligence artificielle, nous avons assisté à l'émergence des attaques basées sur l'IA pour tromper les modèles d'apprentissage automatique [21]. Ces nouveaux types d'attaques sont très subtiles, difficiles à détecter et à contrer, elles sont plus connues sous le nom d'attaques adverses. Lors de la conférence sur le SPAM organisée par le MIT (Massachusetts Institute of Technology), le chercheur Graham-Cumming à clairement montré qu'un filtre antispam à apprentissage automatique pouvait être utilisé pour vaincre un autre filtre antispam à apprentissage automatique en apprenant automatiquement quels mots ajouter à un courriel de spam pour que celui-ci soit classé comme n'étant pas du spam [21],[13],[20]. Également d'autres chercheurs comme Battista Biggio ont montré que les attaques basées sur le

gradient peuvent être utilisées pour tromper des modèles d'apprentissage automatique et d'apprentissage profond [21].

Nous distinguons plusieurs types d'attaques pouvant être classées en deux grands types d'attaques qui sont les attaques à boîte noire et ceux à boite blanche [86]. Dans le contexte des attaques adverses dans l'apprentissage automatique, les attaques à boîte noire sont un type d'attaque dans lequel l'adversaire ne peut obtenir que la sortie pour une entrée fournie, l'adversaire n'a aucune connaissance de la structure ni du paramétrage du modèle. Dans ce type d'attaque, les exemples adverses sont générés soit à partir d'un modèle ou à partir de l'interrogation du modèle d'origine. Contrairement aux attaques à boîte noire, dans les attaques à boîte blanche, l'adversaire a accès aux paramètres du modèle, en plus de cela il est capable d'obtenir les étiquettes pour les entrées fournies [86].

Les principaux algorithmes couramment utilisés pour effectuer des attaques contradictoires sont les suivants :

- FGSM : C'est une méthode simple et efficace utilisant le gradient, principalement utilisée pour générer des images contradictoires. Introduit pour la première fois par Goodfellow dans son ouvrage intitulé "Explaining and Harnessing Adversarial Examples", le fonctionnement de cet algorithme peut être regroupé en 3 grandes étapes [34]. Tout d'abord, il prend en paramètre une image, à partir de laquelle il effectue des prédictions sur l'image à l'aide d'un réseau de neurones convolutionnel. La deuxième étape consiste à calculer le gradient d'une fonction perte (erreur quadratique moyenne ou entropie croisée catégorielle) de la prédiction basée sur l'étiquette de la vraie classe. La dernière étape consiste à calculer le signe du gradient afin de créer une nouvelle image adverse maximisant la perte comme l'illustre la figure 2.18. Le résultat est une image de sortie qui, selon l'œil humain, semble identique à l'original, mais qui amène le réseau neuronal à faire une prédiction incorrecte comme l'illustre la figure 2.18 sur laquelle le modèle a confondu un panda avec un gibbon. le signe de FGSM peut être exprimé par l'équation suivante, $adv_x = x + \epsilon \times sign(\nabla_x J(\theta, x, y))$ ou adv_x est notre image contradictoire de sortie, x est l'image d'entrée originale, y est l'étiquette de l'image d'entrée, ϵ c'est une petite valeur par laquelle nous multiplions les gradients signés afin de garantir que les perturbations sont suffisamment petites pour que l'œil humain ne puisse pas les détecter, mais suffisamment grandes pour tromper le réseau neuronal, θ notre modèle de réseau de neurones et J qui est la fonction de perte.

- PGD : C'est une méthode itérative dans laquelle chaque coefficient est mis à jour

à l'aide de la descente de gradient [51]. Cela facilite la recherche d'une solution dans l'espace contraint. De façon plus explicites, le PGD minimise une fonction soumise à une contrainte. A chaque étape, nous nous déplaçons dans la direction du gradient négatif, puis nous nous "projetons" sur l'ensemble r réalisable. La fonction de perte du PGD est $x^{t+1} = \Pi_{x+S}(x^t) + \alpha sgn(\nabla_x L(\theta, x, y))$. Ce concept est illustré par la figure 2.19, dans laquelle chaque étape de descente de gradient est projetée dans un ensemble réalisable. De cette manière, la solution est régularisée, car la taille de la solution ne peut se situer en dehors de l'ensemble réalisable. Si l'ensemble réalisable est convexe, la solution convergera vers un point optimal à l'intérieur de l'ensemble.

- (C & W) : C'est une attaque puissante qui minimise la perturbation tout en veillant à ce que le classificateur étiquette mal l'exemple adverse. Elle est très efficace, mais plus exigeante en termes de calcul. L'attaque C & W commence par la définition d'une fonction objective, J(x) qui quantifie les objectifs de l'attaque, qui a pour équation $J(x^{'}) = \alpha.dist(x, x^{'}) + loss(f(x^{'}), y_t)$, ou x est l'entrée initiale, $x^{'}$ est l'entrée perturbée, dist(x, x) qui mesure la perturbation, généralement à l'aide de la norme L2 ou L^{∞}, $loss(f(x^{'}), y_t)$ représente la perte de classification erronée du modèle cible f sur l'entrée perturbée par rapport à la classe cible y_t, α et sont des poids qui équilibrent les deux objectifs. L'objectif est généralement une combinaison de deux objectifs contradictoires :

  -Minimiser la perturbation : Pour s'assurer que l'exemple contradictoire reste visuellement similaire à l'entrée originale.

  -Maximiser la confiance dans la classification erronée : Garantir que l'entrée perturbée est mal classée par le modèle cible.

- Attaque par patchs adverses (Adversarial patch attack) : Elles peuvent être utilisés pour attaquer n'importe quelle scène, robustes parce qu'ils fonctionnent sous une grande variété de transformations, et ciblés parce qu'ils peuvent amener un classificateur à produire n'importe quelle classe cible [16]. Ces patchs adverses peuvent être imprimés, ajoutés à n'importe quelle scène, photographiés et présentés à des classificateurs d'images, même lorsque les patchs sont petits, ils amènent les classificateurs à ignorer les autres éléments de la scène et à signaler une classe cible choisie. En analysant la figure 2.20, nous constatons qu'après l'ajout d'un patch contradictoire, la prédiction du modèle passe de "panneau d'arrêt" à "panneau de limitation de vitesse à 80" de manière erronée. Comme nous avons pu le voir sur la figure 2.20, nous créons l'attaque en remplaçant une partie de l'image par notre patch. Par la suite nous masquons le patch, pour lui permettre de prendre n'importe quelle forme. La dernière étape consiste à entraîner le modèle sur une variété d'images, en appliquant une translation, une

mise à l'échelle et une rotation aléatoires sur le patch dans chaque image, en optimisant à l'aide de la descente de gradient. La fonction objective est $\hat{p} = argmax E_{x\ X,t\ T,l\ L}[logPr(\hat{y}|A(p,x,l,t))]$, où X est un ensemble d'images d'apprentissage, T est une distribution sur les transformations du patch et L est une distribution sur les emplacements dans l'image.

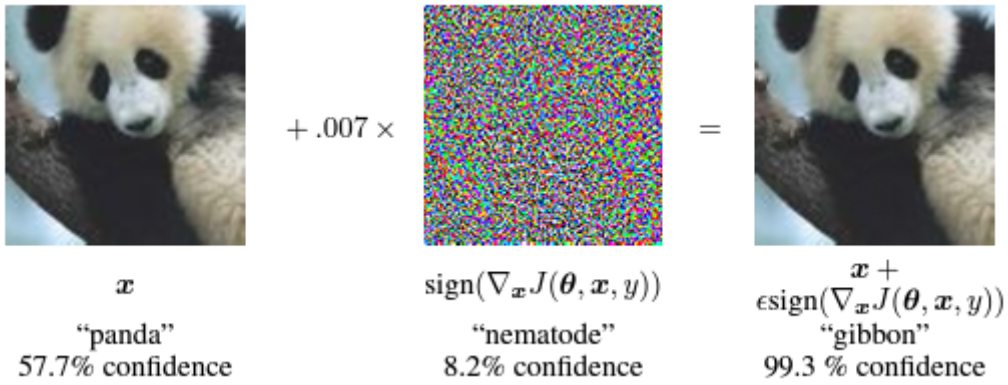


FIGURE 2.18 – FGSM pour la génération d'images contradictoires. Source : Explaining and Harnessing Adversarial Examples

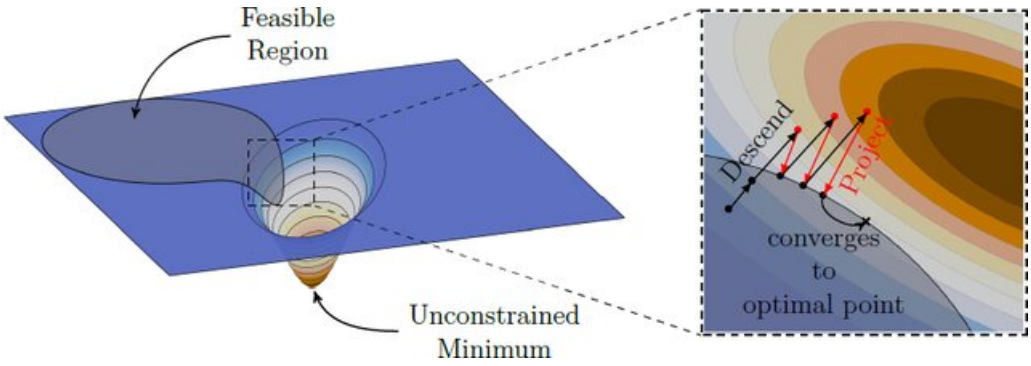


FIGURE 2.19 – Méthode du gradient projeté. Source : researchgate

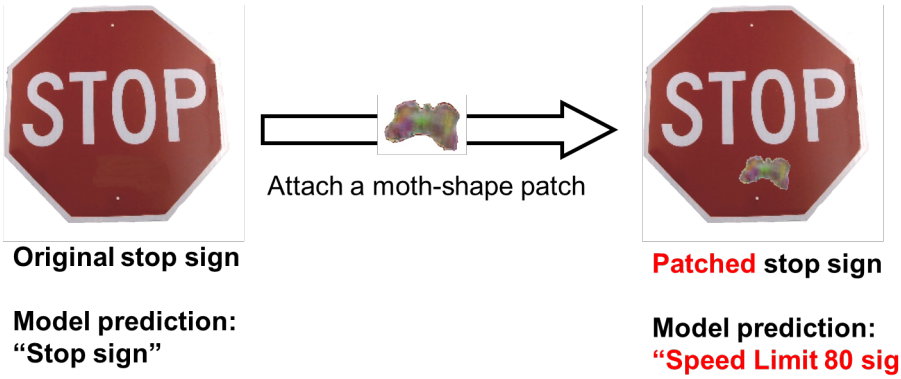


FIGURE 2.20 – un exemple visuel d'attaques par patchs adverses. Source : Yakura et al.

2.14.1 Empoisonnement des données

L'empoisonnement des données consiste à contaminer les données d'entraînement du modèle avec des données conçues pour augmenter l'erreur de la sortie du modèle [86] (figure 2.21). Cette technique peut potentiellement reprogrammer les algorithmes d'apprentissage automatique avec des intentions malveillantes. Ce type d'attaque est de nature de boîte noire, car l'adversaire n'a aucune information sur la structure du modèle. Il est également important de noter qu'il existe plusieurs types d'attaques d'empoisonnement qui sont les suivants :

- ❖ L'empoisonnement d'étiquette : Les adversaires injectent des données mal étiquetées ou malveillantes dans l'ensemble d'apprentissage afin d'influencer le comportement du modèle pendant l'inférence.

- ❖ Empoisonnement des données d'entraînement : Dans le cas de l'empoisonnement des données d'entraînement, l'adversaire modifie une partie importante des données d'entraînement afin d'influencer le processus d'apprentissage du modèle. Les exemples trompeurs ou malveillants permettent à l'attaquant d'orienter la prise de décision du modèle vers un résultat particulier.

- ❖ Attaque par inversion du modèle : Dans les attaques par inversion de modèle, les adversaires exploitent les réponses du modèle d'intelligence artificielle pour en déduire des informations sensibles sur les données sur lesquelles il a été formé. En manipulant les requêtes et en analysant les résultats du modèle, l'attaquant peut extraire des informations privées ou des détails sur l'ensemble de données.

- ❖ Attaques furtives : Dans les attaques furtives, l'adversaire manipule stratégiquement les données d'entraînement pour créer des vulnérabilités difficiles à détecter pendant les phases de développement et de test du modèle. L'attaque vise à exploiter ces faiblesses cachées une fois que le modèle est déployé dans des scénarios réels.

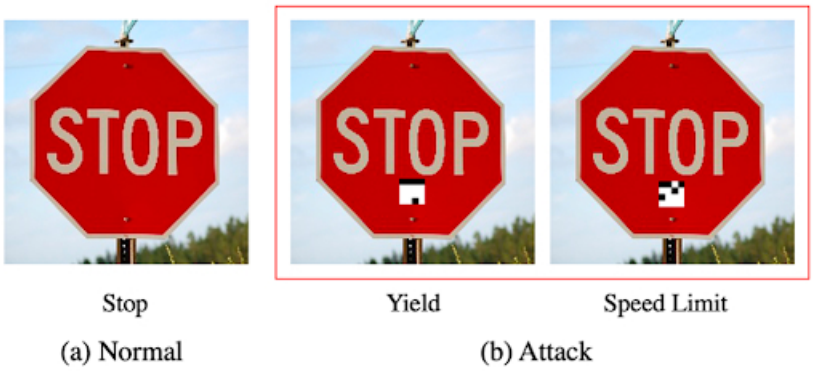


FIGURE 2.21 – Empoisonnement des données. Source : Datascientest

2.14.2 Attaques byzantines

L'attaque byzantine est l'attaque menée par un nœud de confiance qui est devenue rebelle et qui a déjà passé tous les processus d'authentification et de vérification [86]. Lorsqu'un nœud de confiance est pris en défaut, il peut facilement effectuer une attaque DoS sur la couche de contrôle d'accès au support pour empêcher les autres nœuds de communiquer. Ce type d'attaque est très récurrent dans l'apprentissage fédéré comme l'illustre la figure 2.22.

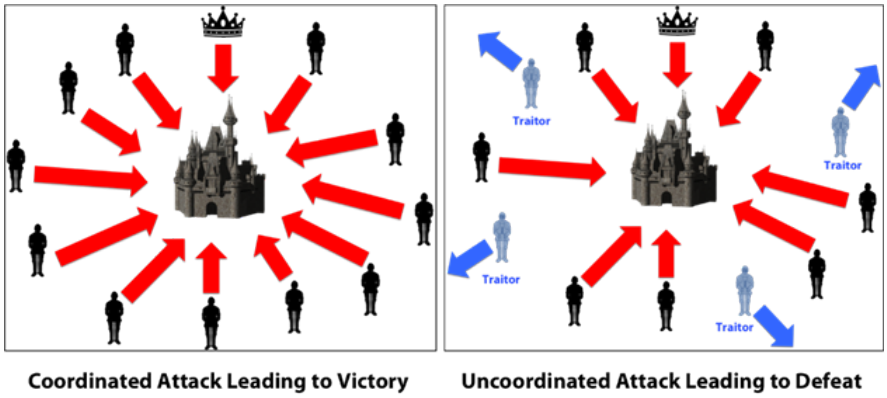


FIGURE 2.22 – Tolérance de panne byzantine pratique dans les systèmes de paiement décentralisés. Source : Captainaltcoin

2.14.3 Evasion

Les attaques par évasion consistent à exploiter l'imperfection d'un modèle formé. Par exemple, les spammeurs et les pirates informatiques tentent souvent d'échapper à la détection en obscurcissant le contenu des spams et des logiciels malveillants. Les échantillons sont modifiés pour échapper à la détection, c'est-à-dire pour être classés comme légitimes (figure 2.23) [86]. Un exemple clair d'évasion est le spam basé sur l'image dans lequel le contenu du spam est incorporé dans une image jointe pour échapper à l'analyse textuelle des filtres antispam. Un autre exemple d'évasion est donné par les attaques par usurpation d'identité contre les systèmes de vérification biométrique.

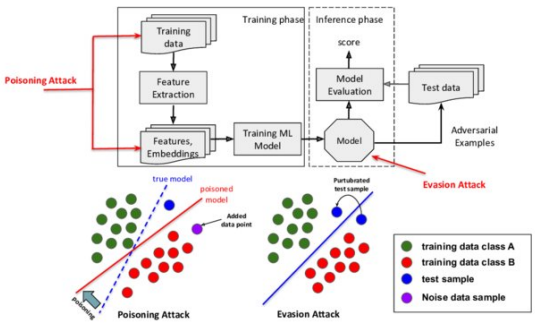


FIGURE 2.23 – Représentation schématique de la distinction entre les attaques par évasion et les attaques par empoisonnement. Source : Researchgate

2.14.4 Extraction du modèle

L'extraction du modèle implique qu'un adversaire sonde un système d'apprentissage automatique en boîte noire afin d'extraire les données sur lesquelles il a été entraîné, cela peut être très dangereux si les données en question sont sensibles ou confidentielles [86], cela est illustré sur la figure 2.24. Dans ce type d'attaque, le but principal est la collecte des données d'entraînement du modèle afin de former un modèle de substitution qui pourra voler la fonctionnalité du modèle cible.

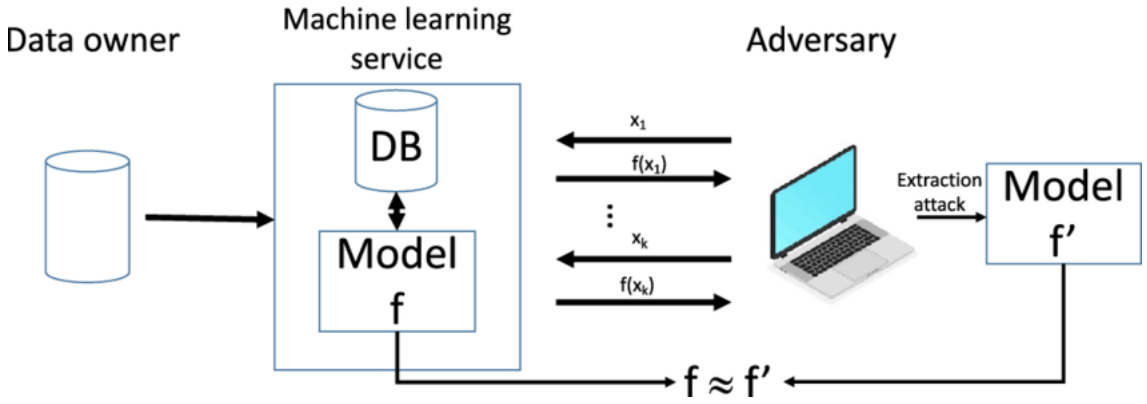


FIGURE 2.24 – Extraction du modèle. Source : Researchgate

2.14.5 Contrer les attaques adverses

Bien que les attaques par apprentissage automatique contradictoire sont très subtiles et difficiles à contrer, cependant certains chercheurs ont proposé des solutions pour contrer ce type d'attaque [21]. Les solutions les plus pertinentes ayant retenu notre attention sont les suivantes :

- L'entraînement contradictoire consiste à entraîner notre modèle à partir d'exemples contradictoires dans l'optique d'améliorer la robustesse du modèle et également sa précision et sa fiabilité.

- Les méthodes d'ensemble permettent une meilleure défense contre les attaques adverses, en effet elle permet d'améliorer la robustesse du système en utilisant un ensemble de classificateurs pour classer de nouvelles données en fonction de la moyenne pondérée de leur prédiction.

- L'une des méthodes les plus efficaces pour prévenir les attaques adverses est de valider et d'assainir les données d'entrée avant de les introduire dans le modèle. Il s'agit de vérifier l'absence d'anomalies, de valeurs aberrantes ou de modifications malveillantes susceptibles d'affecter la performance ou l'intégrité du modèle.

Chapitre 3

Travaux connexes

3.1 Introduction

Avec la numérisation de plus en plus croissante dans notre monde, nous avons assisté à une hausse significative de la fraude en ligne [5]. Parmi les différents types de fraude ayant émergé en ligne, l'une des plus importantes est la fraude sur les cartes bancaires. La fraude sur les cartes bancaires est un type d'usurpation d'identité dans lequel une personne autre que le propriétaire effectue une transaction illégale en utilisant une carte de crédit ou une carte débit ou les détails d'un compte [5],[10]. Cette hausse des fraudes sur les cartes bancaires en ligne s'explique notamment en partie par la mutiliplaction des achats qu'on effectue en ligne, en effet les clients n'ont plus besoin d'avoir de l'argent liquide pour effectuer des achats et ont tendance à privilégier les paiements par carte bancaire. Selon une étude, effectuée par TransUnion (l'une des deux agences d'évaluation du crédit au Canada) conduite d'Avril à Juillet 2023 nous apprenons que 49% des Canadiens ont déclaré avoir été ciblés par un système de fraude récemment [73]. TransUnion a également constaté que les tentatives de fraude numérique, où la transaction provenait du Canada et visait des entreprises mondiales, ont augmenté de 40% en comparant le premier semestre 2022 au premier semestre 2023 [73]. Cette hausse significative des fraudes sur les cartes bancaires se traduit par une perte significative pour les institutions financières et les particuliers pouvant se chiffrer à des milliards de dollars par an. Cela soulève donc la nécessité de la mise en place des systèmes de détection des fraudes ce qui ne sauraient que passer par la recherche, car les fraudeurs sont de plus en plus subtils et évolutifs dans leurs techniques de fraude.

Dans ce chapitre, nous allons procéder à une recherche ainsi qu'une analyse avancée des différentes solutions parut dans des articles de recherche pour résoudre la problématique des fraudes sur les cartes bancaires. Plus concrètement nous nous intéresserons à l'état des connaissances de la recherche dans le domaine de la détection des fraudes sur les cartes bancaires, nous vous présenterons une synthèse des différentes solutions innovantes permettant une lutte efficace contre les fraudes sur les cartes bancaires.

3.2 Approches basées sur l'apprentissage supervisées

Les techniques de détection d'anomalies supervisées partent de l'hypothèse que l'ensemble de données utilisé est constitué d'instances étiquetées qui appartiennent soit à une classe normale, soit à une classe anormale. La plupart des approches de cette catégorie construisent un modèle prédictif pour les classes normales et anormales, et les nouvelles données non vues peuvent être classées en les comparant au modèle déterminé. Les résultats de nos recherches concernant l'utilisation de l'apprentissage supervisé pour la détection des fraudes sont contenus dans les tableaux 3.1 et 3.2.

La première solution à laquelle nous allons nous intéresser est celle des auteurs Sahin et Duman qui dans leur article de 2011, préconisent l'utilisation des algorithmes SVM et arbre de décision pour la détection des fraudes sur les cartes de crédit [39]. La première étape est la phase de prétraitement qui consiste à déterminer les caractéristiques les plus pertinentes pour distinguer les transactions frauduleuses des transactions légitimes. Ensuite pour résoudre le déséquilibre présent dans leur jeu de données, les auteurs ont opté de sous-échantillonner les transactions normales. Ainsi donc, des échantillons stratifiés sont ensuite constitués à partir des transactions légitimes en utilisant ces variables, puis combinés avec les transactions frauduleuses. Les transactions sont par la suite envoyées aux algorithmes SVM et arbre de décision pour la classification. Les résultats des expérimentations montrent que l'arbre de décision est plus performant que le modèle SVM. L'application du sous-echantillonage entraine une perte d'information, ce qui peut impacter la performance des différents algorithmes. Également il aurait été judicieux de penser à appliquer des techniques génération de données synthétiques dans l'optique d'augmenter la performance des différents modèles.

D'autres auteurs comme Rtayli et Enneya proposent une méthode améliorée de détection des fraudes sur les cartes de crédit [39]. Tout d'abord ils utilisent Random Forest pour la sélection des caractéristiques qui sont les plus pertinentes dans l'ensemble de données. Ces caractéristiques sont par la suite envoyées à un modèle SVM qui est chargé d'identifier les transactions frauduleuses. Les métriques de performance considérées par l'auteur sont "Accuracy", la précision, le rappel, l'aire sous la courbe (AUC). Le jeu de données choisi par les auteurs est composé de 284,807 transactions parmi lesquelles 492 sont des transactions frauduleuses. La solution présente un taux d'Accuracy de 95.18% et un taux de rappel de 87%. Les conclusions des expérimentations montrent que la combinaison de Random Forest et SVM produit une bonne précision avec moins d'alerte de faux positifs.

Les auteurs Behera et Panigrahi se sont intéressés à l'utilisation des Perceptrons multicouches (MLP) et d'un modèle de détection de regroupement flou (Fuzzy c-means) pour une détection efficace des fraudes sur les cartes de crédit [39]. Le Fuzzy c-means

(FCM) est un regroupement qui permet à chaque point de données d'appartenir à plus d'un groupe. Les transactions sont regroupées par le Fuzzy c-means sur la base de deux caractéristiques qui sont le montant de la transaction et l'article acheter. Un score de suspicion est alors calculé sur la base de la distance euclidienne de la transaction par rapport au centroïde du groupe. Le score de suspicion est ensuite comparé à des seuils supérieurs et inférieurs déterminés expérimentalement. S'il se situe dans le seuil inférieur, il s'agit d'une transaction normale, s'il se situe dans le seuil supérieur, il s'agit d'une transaction frauduleuse et s'il se situe entre les deux seuils, la transaction est considérée comme suspecte, et dans ce cas c'est le MLP qui est chargé d'effectuer la classification. Cinq couches cachées ont été utilisées pour le MLP, ce qui a permis d'améliorer les performances au prix d'une augmentation du temps de calcul pour l'apprentissage. Les résultats ont montré que la méthode proposée permettait d'obtenir un taux de vrais positifs de 93,90% et un taux de faux positifs ou de fausses alarmes de seulement 6,10%. Les suggestions futures dans la littérature suggèrent de prendre en compte davantage de caractéristiques par le système pour améliorer le modèle. On note que dans cette solution, les auteurs ne résolvent pas la problématique du déséquilibre des données. Outre cela les expérimentations sont effectuées sur un seul jeu de données, cela ne permet pas de connaître comment la solution performerait sur d'autres jeux de données.

Les auteurs Cheng, Tu et Zhang dans leurs articles se sont intéressés à l'utilisation des réseaux neuronaux convolutionnels pour la détection des fraudes sur les cartes de crédit [39]. Cette méthode a été proposée en raison de la tendance des réseaux MLP à effectuer un surapprentissage (overfitting). Pour effectuer leurs expérimentations, les auteurs ont choisi un ensemble de données de cartes de crédit étiquetées provenant d'une banque commerciale. Ce jeu de données étant privé, alors nous avons peu d'informations sur le contenu du jeu de données. Tout d'abord les auteurs, ont procédé à la transformation des caractéristiques pour capturer les représentations temporelles qui sont présentes dans les données. De nouvelles caractéristiques ont été dérivées des données brutes, telles que le montant moyen de la transaction, les différences entre un montant actuel et le montant moyen, ainsi que d'autres caractéristiques générées. Les auteurs proposent une nouvelle caractéristique pour décrire la relation entre le montant de la transaction d'un utilisateur et le montant total de la transaction sur une période, connue sous le nom d'entropie de négociation, qui a été mise en œuvre dans le modèle. Ensuite, les vecteurs de caractéristiques unidimensionnels d'origine ont été transformés en une matrice de caractéristiques dans laquelle les lignes représentent les différentes caractéristiques et les colonnes les différentes fenêtres temporelles. Les résultats de l'expérimentation montrent que le réseau neuronal convolutionnel performe mieux que le MLP et SVM.

Les auteurs Vinay Arora , Rohan Singh Leekha,Kyungroul Lee et Aman Kataria dans leur article intitulé "Facilitating User Authorization from Imbalanced Data Logs of Credit Cards Using Artificial Intelligence" propose de détecter les fraudes grâce aux des cartes de crédit [9]. Les auteurs pour l'expérimentation de leurs solutions ont choisi trois jeux de données qui sont fortement déséquilibrés [44], [47], [45]. Ces trois jeux de données sont publics et disponibles sur Kaggle, le premier jeu de données est de petite taille avec 3075 transactions et 11 colonnes, sur ces 3075 transactions seules 448 sont frauduleuses. Le deuxième jeu de données est de taille moyenne et contient 30000 transactions dont 6636 présente un défaut de paiement. Le troisième jeu de données est de grande taille avec 284807 transactions dont seulement 492 sont des transactions frauduleuses c'est à dire que seulement 0.172% des transactions sont frauduleuses. Ce jeu de données a également comme particularités d'avoir subit l'ACP pour des raisons de confidentialités et les colonnes ont été renommés de V_1 a V_{28}, seules les colonnes temps et montant n'ont pas subit le ACP. Cette solution peut être résumée en deux grandes étapes principales. La première étape consiste à appliquer un sous-échantillonnage aléatoire et un renforcement à l'aide d'un arbre de décision, pour cela les auteurs ont utilisé l'algorithme RUSBoost. L'algorithme RUSBoost est une combinaison de RUS (sous-échantillonnage aléatoire) et de la procédure de renforcement standard AdaBoost. La dernière étape consiste à utiliser l'algorithme SVM dans le processus de classification. Les résultats obtenus sont par la suite comparés aux algorithmes d'arbres de décision, régression logistique, perception multicouche, KNN, Random Forest, AdaBoost et SVM, montrant que la solution des auteurs présente le pourcentage le plus élevé du ratio de vrai positif avec 96.3% sur le premier jeu de données et 99.6% sur les deux autres jeux de données. Également le modèle présente la précision (94.2%, 95.7%, 95.7%) et le ScoreF1 (88.6%, 97.6%, 97.6%) le plus élevé sur les trois jeux de données. Bien que la solution présentée par les auteurs soit pertinente, elle reste sujette à de nombreuses critiques, tout d'abord nous notons l'absence d'une phase de prétraitement des trois jeux de données alors qu'il est crucial de préparer les données en vue de leur utilisation par les algorithmes d'apprentissage automatique. Également l'utilisation du sous-échantillonnage peut entraîner une perte d'information précieuse provenant de la classe majoritaire et cela peut conduire à des modèles moins précis. Enfin il aurait été judicieux d'explorer des techniques de génération de données.

Similairement à la première solution , les auteurs Afriyie, Jonathan Kwaku Tawiah, Kassim Pels dans leur article intitulé "A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions" paru en 2023 proposent une solution efficace pour contrer les fraudes sur les cartes de crédit [2]. Cette solution repose sur l'utilisation des algorithmes d'apprentissage automatique que nous vous présenterons. L'ensemble de données choisi par l'auteur est un ensemble de transactions de cartes de crédit simulées par un générateur effectué le 1er janvier 2020 et le 31

décembre 2020 contenant à la fois des transactions normales et frauduleuses d'un total de 555 719 lignes d'observation, dont 23 colonnes [68]. Ce jeu de donnée a comme particularités d'être fortement déséquilibré. La première étape est la phase de prétraitement des données, phase dans laquelle les données ont été nettoyées et formatées pour éliminer les valeurs manquantes. Également, une mise à l'échelle est effectuée pour faire en sorte que toutes les caractéristiques (colonnes) aient des valeurs comprises entre 0 et 1. Par la suite pour résoudre le déséquilibre présent dans l'ensemble de données, les auteurs ont appliqué la technique de sur échantillonnage des minorités synthétiques SMOTE plus particulièrement en sous-échantillonnage cela a eu pour effet de réduire les données de la classe majoritaire de sorte qu'elles soient à peu près égales à ceux de la classe minoritaire. La régression logistique, le Random Forest, l'arbre de décision sont utilisés pour la détection des fraudes. Les résultats des expérimentations des auteurs montrent que l'algorithme Random Forest est celui qui a la meilleure performance avec une valeur d'AUC de 98.9 %, suivi du modèle Decision Tree avec une valeur d'AUC de 94.5%. Ainsi donc les auteurs tirent comme conclusion que Random Forest est plus utile pour prédire les transactions frauduleuses, car le ratio taux de vrais positifs et le taux de faux positifs sont proches de 1. Le modèle de régression logistique présente une valeur d'AUC de 87.2%. Cette solution, bien que pertinent reste sujet à certaines critiques, tout d'abord l'utilisation du sous-échantillonnage peut entraîner une perte d'information précieuse provenant de la classe majoritaire, cela peut conduire à des modèles moins précis. Également il aurait été plus judicieux d'utiliser plusieurs jeux de données, car cela exposerait les modèles à de plus larges éventails de variations, ce qui leur permet d'apprendre des représentations de caractéristiques plus complètes et plus discriminantes.

Les auteurs Suhas Jain, Rakesh, Pranavi, Bale, Lahar dans leur article intitulé "A Novel Approach in Credit Card Fraud Detection System Using Machine Learning Techniques" paru en 2021 proposent une solution novatrice pour une détection efficace des fraudes sur les cartes de crédit [69]. Les auteurs pour leur expérimentation ont choisi le jeu de données qui est composé de 2,84,000 transactions dont 492 sont des transactions de fraude, ce jeu de données en plus d'être déséquilibrés a comme particularité d'avoir subi l'analyse en composante principale pour des raisons de confidentialité [45]. La phase de prétraitement est la première étape de leur solution, phase dans laquelle ils sélectionnent au hasard 492 transactions sur les transactions normales afin d'obtenir un ratio de 50 :50. Également durant cette phase de prétraitement, ils suppriment toutes les valeurs aberrantes qui sont présentes dans le jeu de données. Par la suite une réduction du nombre de variables sera effectuée par l'utilisation des techniques de réduction de dimensionnalité (SNE, techniques SVD tronqué). La dernière étape consiste à l'application de deux algorithmes de classification que sont la régression logistique et l'arbre renforcé par le gradient (XGBoost). Selon les

auteurs l'arbre renforcé par le gradient (XGBoost) est l'algorithme qui performe le mieux. Plusieurs critiques peuvent être portées, en sélectionnant uniquement 492 transactions normales cela entraîne une perte d'information qui peut impacter la performance des algorithmes d'autant plus que la quantité de données pour l'entraînement des modèles n'est pas assez grande (492 transactions normales et 492 transactions frauduleuses). Également en réduisant la dimensionnalité avec SNE, cela entraîne davantage une perte d'information tout comme le sous-échantillonnage.

Ref	**Technique**	**Jeu de données**	**Pré traitement**	**Métriques**	**Résultat**
[39]	SVM, Arbre de décision, Sous-échantillonnage	National bank's credit card data warehouses(jeu de données privées).	√	Accuracy	Les résultats ont montré que les modèles d'arbres de décision ont surpassé les modèles SVM sur l'ensemble de test.
[39]	Random Forest(sélection des caractéristiques), SVM	European cardholders credit cards 2013.	√	Accuracy, Précision, AUC	Système capable de détecter efficacement les cas de fraude
[39]	MLP,Fuzzy c-means(FCM)	Jeu de données privées.		Accuracy, Taux de vrais positifs, Taux de faux positifs.	La combinaison du FCM et du MLP ont t permis d'obtenir une "Accuracy" de 93.90%, avec un taux de fausses alarmes de seulement 6.10 %.

TABLE 3.1 – Comparaison des solutions de détection des fraudes sur les cartes bancaires via l'apprentissage supervisé Partie 1

Ref	**Technique**	**Jeu de données**	**Pré traitement**	**Métriques**	**Résultat**
[39]	Transformation des caractéristiques, CNN	Jeu de données privées(banque commerciale).	√	Efficacité du calcul, Rappel, Précision, F1Score	CNN a obtenu un scoreF1 de 33% et a surpassé le MLP.
[9]	RUSBoost, Classificateur (SVM)	-Joshi S., Abstract dataset for credit card fraud detection, 2020. -Learning U. M., Default of credit card clients dataset, 2016. -ULB M. L. G., Credit card fraud detection, 2018.		Matrice de confusion, Précision, ScoreF1, ROC, AUPR, Sensibilité et Spécificité	La solution proposée performe mieux que l'arbre de décision, la régression logistique, perception multicouche (MLP), KNN, Adaboost et RandomForest.
[2]	SMOTE sous-échantillonnage, Classificateur (régression logistique, Random Forest, Arbre de décision)	Simulated Credit Card Transactions generated using Sparkov. 2020.	√	Accuracy, ScoreF1, Rappel, Précision, Spécificité, AUC	AUC : Random Forest 98.9 %, Decision Tree 94.5%, Régression logistique 87.2%.
[69]	SNE, SVD tronquées, Classificateur(XGBoost, Régression logistique)	European cardholders credit cards 2013.	√	Accuracy, Précision, ScoreF1	L'algorithme XGBoost performe mieux que la régression logistique.

TABLE 3.2 – Comparaison des solutions de détection des fraudes sur les cartes bancaires via l'apprentissage supervisé Partie 2

3.3 Approches basées sur l'apprentissage non supervisées

L'apprentissage non supervisé ne nécessite pas d'étiquette dans l'ensemble de données. Les méthodes non supervisées supposent implicitement que les événements anormaux sont beaucoup moins fréquents que les événements normaux dans l'ensemble de test des données. Dans le cas contraire, les taux de fausse alarme de ces techniques seront plus élevés que prévu. Ainsi nous avons exploré l'état de l'art présentant des solutions d'apprentissage non supervisées pour la détection des fraudes sur les cartes de crédit, les résultats de nos recherches sont contenus dans les tableaux 3.3 et 3.4. La première solution que nous allons explorer est celle des auteurs Ounacer, Bour, Oubrahim, Ghoumari et Azzouazi qui dans leur article se sont intéressés à l'utilisation de l'algorithme Isolation Forest (IF) pour la détection des fraudes sur les cartes de crédit [39]. Isolation Forest est un algorithme permettant d'exploiter le sous-échantillonnage pour créer un algorithme ayant une faible complexité temporelle linéaire constante et un faible besoin en mémoire. Isolation Forest est également capable de traiter des ensembles de données massives et de résoudre des problèmes à haute dimension comportant de nombreuses caractéristiques non pertinentes. La phase d'entraînement consiste à construire un modèle Isolation Forest à l'aide d'arbres avec des sous-échantillons de l'ensemble d'apprentissages. La phase de test fait ensuite passer chaque échantillon par le modèle afin de calculer le nombre de divisions nécessaires dans tous les arbres pour isoler cette observation et renvoyer un score d'anomalie entre 0 et 1, où 1 indique une fraude et où le seuil est fixé à 0,5 pour classer les échantillons. La méthode proposée a été comparée à d'autres modèles non supervisés tels que OCSVM et k-means clustering en utilisant l'ensemble de données des cartes de crédit européennes contenant 284 807 transactions avec seulement 492 transactions frauduleuses. Les résultats ont montré que l'Isolation Forest était le plus performant, avec une précision(Accuracy) et une aire sous la courbe (AUC) de 95,12% et 91,68%, respectivement. L'algorithme de regroupement k-means arrive en deuxième position avec 90,12% et 51,91% pour la précision(Accuracy) et l'AUC, respectivement.

D'autres auteurs se sont quant à eux intéressés à l'utilisation des réseaux de neurones non supervisés pour la détection des fraudes sur les cartes de crédit. C'est ainsi donc que les auteurs Zaslavsky et Strizhak se sont intéressés à l'utilisation des réseaux de neurones SOM pour la détection des fraudes sur les cartes de crédit [39]. SOM est un type de réseau neuronal utilisant l'apprentissage non supervisé qui configure les neurones du réseau en fonction de la structure topologique des données d'entrée. Ce processus, connu sous le nom d'auto-organisation, ajuste de manière itérative les poids des neurones afin d'obtenir une approximation des données d'entrée, ce qui permet de regrouper et de profiler l'ensemble des données. Les neurones d'un SOM sont disposés dans une structure

matricielle qui fait correspondre les entrées d'un espace à haute dimension au réseau bidimensionnel de neurones. Ce mappage est conçu pour modéliser les vecteurs d'entrée similaires comme des neurones plus proches les uns des autres dans la matrice résultante, ce qui permet de visualiser l'entrée. Diverses mesures de distance peuvent être utilisées au cours du processus d'apprentissage itératif pour regrouper les nœuds, telles que la distance euclidienne, la distance de Manhattan. Après l'entrainement, les données sont classées en ensembles légitimes ou frauduleux par auto-organisation, et toute nouvelle transaction ultérieure subit le même processus avant d'être introduite dans le SOM. Ces nouvelles transactions sont classées comme légitimes ou frauduleuses selon qu'elles se situent ou non en deçà d'un seuil spécifique de valeur. Plutôt que d'avoir des classifications binaires pour les étiquettes frauduleuses et légitimes, les méthodes SOM peuvent également avoir plus d'un groupe représentant chaque classe. Les auteurs ont montré que la précision(Accuracy) de la méthode augmentait avec la taille de l'ensemble de données, en particulier pour les cartes de crédit examinées les plus déséquilibrées.

Un Autoencodeur (AE) est un type de réseau d'apprentissage profond non supervisé dont la structure est symétrique, avec moins de nœuds dans les couches intermédiaires [39]. Il comporte une section qui code les entrées dans une représentation de dimension inférieure et une autre section qui décode ou reconstruit à nouveau cette entrée. L'Autoencodeur s'avère une alternative crédible pour la détection des fraudes, car il permet d'apprendre un codage réduit des données de manière efficace, puis de le reconstruire. C'est pour cela que les auteurs Kazemi et Zarrabi ont proposé dans leurs articles l'utilisation des Autoencodeurs pour la détection des fraudes sur les cartes de crédit. Les auteurs suggèrent l'utilisation de deux autoencodeurs avec des configurations différentes. Le premier Autoencodeur comporte 20 neurones dans la couche d'entrée, puis 15, 10 et 5 neurones dans les couches cachées suivantes. Le deuxième Autoencodeur comporte 20 neurones dans la couche d'entrée et 30, 50 et 100 neurones dans les couches suivantes. Les deux modèles des Autoencodeurs ont été comparés à un SOM sur un ensemble de données de cartes de crédit allemandes comprenant 1000 échantillons. Le deuxième Autoencodeur est celui qui performe le mieux avec une précision (Accuracy) de 84.1%, néanmoins SOM a surpassé de 0.8% le premier Autoencodeur avec une précision de 82.4%. Dans l'expérimentation des auteurs, nous constatons l'utilisation d'un seul jeu de données, de surcroit le jeu de données choisi est de petite taille ce qui peut impacter négativement la performance du modèle.

Moschini, Houssou, Bovay, et Robert-Nicoud se sont intéressés à l'utilisation du modèle Arima pour la détection des fraudes sur les cartes de crédit [39]. Le modèle ARIMA est une technique utilisée pour les séries temporelles, c'est un modèle simple qui facilite la différenciation des points de données dans les séries temporelles [39]. Dans le cadre de la détection des fraudes, les séries temporelles sont un outil utile pour traiter

les caractéristiques agrégées produites par l'agrégation, qui est une méthode permettant de dériver des nouvelles caractéristiques dans nos données, ce qui pourraient être plus utile au modèle. Les auteurs proposent ce modèle pour remédier à la nature déséquilibrée des ensembles de données, également ce modèle prend en compte l'évolution des comportements et des modèles de dépenses au fil du temps. Le modèle est d'abord calibré sur le nombre quotidien de transactions légitimes afin d'apprendre les comportements des clients en matière de dépenses. Ensuite, des fenêtres temporelles glissantes sont utilisées pour prédire les transactions frauduleuses à partir de l'ensemble de tests. Les transactions frauduleuses ont été signalées sur la base de leur score standard calculé (également connu sous le nom de score Z) plus significatif qu'un seuil. Les expérimentations ont été effectuées sur le jeu de données fournie par une par une société de prévention des fraudes connue sous le nom de NetGuardians SA. Ce jeu de données contient les informations sur les transactions par carte de crédit de 24 titulaires de cartes entre juin 2017 et février 2019, il est également fortement déséquilibré avec les transactions frauduleuses qui représentent 0.76% de toutes les transactions. Le modèle ARIMA a été mis en œuvre et comparé à d'autres modèles tels que les diagrammes en boîte (box plots), le LOF, l'Isolation Forest et k-means. Les résultats expérimentaux montrent que le modèle ARIMA a affiché la précision et le scoreF1 les plus élevés, soit 50% et 55,56%, respectivement. L'algorithme k-means, cependant, a été le plus performant en termes du score de rappel. Dans l'ensemble, le modèle le plus mauvais est LOF, avec une précision de 8.4% et un scoreF1 de 14.04%.

Ref	**Technique**	**Jeu de données**	**Pré traitement**	**Métriques**	**Résultat**
[39]	Isolation Forest	European cardholders credit cards 2013.		Accuracy, AUC.	Isolation Forest est plus performant que les algorithmes OCSVM et K-means.
[39]	SOM (Entraînement et classification des données (légitimes ou frauduleux) par auto-organisation)	Jeux de données non spécifiés.	√	Accuracy	La précision augmente avec la taille de l'ensemble de données.

TABLE 3.3 – Comparaison des solutions de détection des fraudes sur les cartes bancaires via l'apprentissage non supervisé Partie 1

Ref	Technique	Jeu de données	Pré traitement	Métriques	Résultat
[39]	Autoencodeurs	German credit card dataset.		Accuracy .	Le deuxième Autoencodeur est celui qui performe le mieux avec une précision (Accuracy) de 84.1%, néanmoin SOM a surpassé de 0,8% le premier Autoencodeur avec une précision de 82.4%.
[28]	ARIMA Model.	Jeu de données privées de NetGuardians.		Score Z,Accuracy, ScoreF1.	Le modèle ARIMA présente une Accuracy de 50% et un scoreF1 de 55.56% .

TABLE 3.4 – Comparaison des solutions de détection des fraudes sur les cartes bancaires via l'apprentissage non supervisés(partie 2)

3.4 Approches basées sur l'apprentissage semi-supervisé

Nous allons nous intéresser aux approches utilisant l'apprentissage semi-supervisé pour la détection des fraudes. En effet l'apprentissage semi-supervisé reste une alternative crédible pour la détection des fraudes, cela a conduit de multiples chercheurs à effectuer des recherches sur des solutions éventuelles basées sur l'apprentissage semi-supervisé.

Les premières techniques auxquelles nous allons nous intéresser sont celles des modèles de Markov cachés. Les modèles de Markov cachés se composent d'un ensemble fini d'états régis par un ensemble de probabilités de transition [39]. Chaque état possède également une distribution de probabilités associée responsable de la génération d'un résultat ou d'une observation. La première solution à laquelle, nous allons nous intéresser est celle des auteurs Srivastava, Kundu, Sural, Majumdar qui, dans leur article parut en 2008, propose une solution basée sur l'utilisation des modèles de Markov

cachés multimoniaux pour une détection efficace de la fraude sur les cartes de crédit [39]. Dans cette technique, les transactions normales d'un individu sont utilisées pour entraîner les modèles de Markov cachés multinomiaux pour chaque possesseur d'une carte de crédit. Chaque transaction entrante d'un individu est comparée à son modèle de transaction normale préalablement entraîné, toute transaction non acceptée par le modèle de Markov caché multinomial avec une probabilité suffisamment élevée est automatiquement considérée comme frauduleuse. Comme nous pouvons le voir dans le tableau 3.5, la première étape de cette solution consiste à quantifier pour chaque possesseur de carte de crédit la valeur de ses achats en fourchettes de prix. Par la suite les auteurs utilisent l'algorithme k-means pour déterminer les fourchettes de dépenses en trois catégories qui sont faibles, moyens et élevé. Les auteurs ont utilisé des données générées par un simulateur pour l'entraînement et l'évaluation de la performance de leur modèle, ce choix s'explique par la difficulté d'obtenir des données d'institutions qui hésitent à les partager [39]. Basés sur les différents tests effectués, il ressort qu'une longueur de séquence 100 et 5 états est la meilleure configuration. Les auteurs notent également une augmentation de la performance et de la complexité de calcul en fonction de la longueur de la séquence. Selon les auteurs, la méthode qu'ils proposent est plus performante que les méthodes proposées dans les littératures antérieures. Cette méthode présente un taux "Accuracy" de près de 80% en moyenne sur une grande variété de données d'entrée et il a été démontré qu'il était extensible pour traiter de grands volumes de transaction. Cette solution peut présenter une latence, ce qui ne la rend pas adaptée pour une détection en temps réel. Les auteurs n'ont également pas testé la robustesse de leurs modèles contre les attaques malicieuses.

D'autres auteurs comme Bhusari et Patil ont également vérifié la performance des modèles de Markov cachés pour la détection des fraudes sur les cartes de crédit, pour cela ils ont recréé le modèle proposé par Srivastava et ils ont obtenu des résultats similaires sur un ensemble de données simulées [39]. Les résultats obtenus sont cohérents avec les résultats de la littérature précédente, cela démontre la faisabilité et la performance des modèles de Markov cachés.

Les auteurs Lucas, Portier, Laporte dans leur article paru en 2020 proposent une solution reposant sur la combinaison d'un modèle de Markov caché et d'un classificateur de random Forest pour une détection efficace de la fraude [39]. Dans cette technique, la chaîne de Markov caché est utilisée pour la création des caractéristiques séquentielles pour décrire les dépendances temporelles entre les transactions des titulaires de cartes. Par la suite, Random Forest est utilisé pour détecter les transactions frauduleuses sur la base des caractéristiques séquentielles générées par le modèle de Markov caché. La première étape de cette méthode consiste à comparer les probabilités des séquences de transactions légitimes et celles frauduleuses. Suite à cela, les caractéristiques décrivant

les séquences de transactions des titulaires de carte de crédit et des terminaux des commerçants vont être créées. En combinant ces perspectives, on obtient huit ensembles de séquences produites à partir de l'ensemble d'entraînement des données, chaque séquence est entraînée sur un modèle de Markov caché. Chaque modèle de Markov caché associe une valeur de vraisemblance à une transaction sur la base des transactions précédentes, ces valeurs sont par la suite utilisées comme caractéristiques supplémentaires pour le classificateur Random Forest. Les résultats des expérimentations montrent que l'AUC de la courbe précision-rappel de la nouvelle méthode a augmenté de 18,1% pour les transactions effectuées en face-à-face et de 9,3% pour les transactions d'e-commerce.

Une autre technique d'apprentissage semi-supervisé à laquelle nous allons nous intéresser est celle des réseaux génératifs adverses (GAN), cette technique s'avère très utile pour la détection des fraudes sur les cartes de crédit. Ainsi donc, plusieurs chercheurs ont exploré leur utilisation pour la détection des fraudes sur les cartes de crédit. Les auteurs Chen, J., Shen, Y., et Ali, R dans leurs articles parus en 2018 propose une solution pour la détection des fraudes se basant sur l'utilisation d'un "Sparse Autoencoder" et d'un réseau génératif adverse [39]. Dans cette technique, le "Sparse Autoencoder" est utilisé pour obtenir les représentations des transactions normales pour pouvoir entraîner le réseau génératif adverse. L'architecture d'un GAN est constituée d'un générateur et d'un discriminateur. Le discriminateur est entraîné pour distinguer les transactions authentiques réelles des transactions authentiques fictives, qui sont produites par le générateur formé sur les représentations normales apprises et produites par le "Sparse Autoencoder". En comparant la méthode que proposent les auteurs à d'autres solutions comme OCSVM proposées par Tax et Duin dans (Tax et Duin, 2001), ainsi que les processus gaussiens à une classe (OCGP), proposés par Kremmler, il ressort que cette méthode est la plus performante avec un scoreF1 de 87.36% et une précision de 97.59%. Cependant, OCSVM présente un taux de rappel plus élevé qui est de l'ordre de 95.11% alors que la solution proposée par les auteurs présente un taux de rappel de 79.37%. Le discriminateur du GAN a été testé avec et sans les caractéristiques du "Sparse Autoencoder". Une amélioration des performances a été prouvée avec une augmentation de 10.18% de la précision de 87.41% à 97.59% et une augmentation de 3.89% du score F1 de 83.47% à 87.36%.

Les auteurs Tanaka et Aranha quant'à eux se sont intéressés à l'utilisation des réseaux génératifs adverses pour la détection des fraudes [39]. Les GAN peuvent générer des données synthétiques qui s'avèrent très utiles pour la résolution des problèmes de classification dans lesquels le jeu de données est déséquilibré. La capacité du GAN à générer de nouvelles données synthétiques est similaire à d'autres techniques de génération de données synthétiques comme SMOTE ou l'échantillonnage synthétique

adaptatif (ADASYN). Tout d'abord, la classe minoritaire (transaction frauduleuse) est séparée de l'ensemble de données, suite à cela plusieurs GANS avec un nombre et une taille variable de couches cachées sont entraînées uniquement sur ce sous-ensemble de données (classe minoritaire). Les GAN ont été utilisés par les auteurs pour générer de nouvelles données synthétiques afin d'augmenter le jeu de données jusqu'à ce qu'ils soient équilibrés. Après avoir équilibré le jeu de données, les auteurs ont entraîné un classificateur d'arbre de décision sur le jeu de données augmenté (jeu de données initial combiné aux données synthétiques générées par le GAN). La méthode proposée a été comparée à d'autres modèles d'arbre de décision utilisant SMOTE ou ADASYN. Pour toutes les méthodes avec suréchantillonnage, les auteurs notent une amélioration d'au moins 30% du score de rappel.

D'autres auteurs comme Fiore, De Santis ont également exploré l'utilisation des GAN pour améliorer l'efficacité de la classification dans la détection de la fraude sur les cartes de crédit. Ainsi donc, dans leur article les auteurs ont choisi de combiner le GAN avec un classificateur MLP [39]. Le classificateur MLP a été entraîné sur 2/3 de l'ensemble du jeu de données originales afin de déterminer l'ensemble des hyperparamètres optimaux qui permettent d'obtenir les meilleures performances sur l'ensemble de tests. Ensuite le GAN est entraîné uniquement sur les échantillons frauduleux afin de générer des données synthétiques de fraudes. Un autre MLP est entrainé sur le jeu de données augmenté en utilisant les paramètres du premier MLP entrainé. Les résultats des expérimentations montrent une amélioration de la performance du modèle quand il est entraîné sur le jeu de données augmenté, en effet le score du rappel s'est amélioré, passant de 70,23% dans l'ensemble de données déséquilibré à 73,03% dans le jeu de données équilibrées. Cette amélioration a été contrebalancée par une diminution négligeable de la spécificité de 0,004% par rapport à 99,9998 %, ce qui correspond à l'augmentation du nombre de faux positifs. Les auteurs pour la génération des données utilisent GAN, mais il aurait également été judicieux d'explorer d'autres variations de GAN et comparer leurs performances en termes de qualité de données générées. Les auteurs ne comparent pas leur l'approche avec d'autres approches pour une meilleure évaluation.

Le GAN de Wasserstein (WGAN) a été proposé par Arovsky qui a utilisé la distance Earth Mover (EM) comme mesure de l'erreur, contrairement au GAN classique l'architecture du WGAN ne nécessite pas une conception minutieuse [53]. Les auteurs Ba et H dans leurs articles ont proposé l'implémentation du WGAN au lieu du GAN, car le WGAN est plus stable dans sa phase d'entraînement et capable de produire des transactions frauduleuses plus réalistes quand il est entraîné sur la classe minoritaire [39]. Le WGAN a été utilisé pour suréchantillonner la classe minoritaire et équilibrer l'ensemble de données. Cet ensemble de données a ensuite été utilisé pour améliorer les

performances d'un classificateur de régression logistique. Les résultats des expérimentations montrent que le WGAN s'avère plus performant que le GAN classique.

Ref	**Technique**	**Jeu de données**	**Pré traitement**	**Métriques**	**Résultat**
[39]	K-means, Modèles de Markov cachés.	Simulateur de données non spécifié.		Performance, Complexité de calcul, Accuracy.	"Accuracy" de près de 80% en moyenne sur une grande variété de données d'entrée.
[39]	Modèle de Markov caché, Random forest.	Cartes de crédit belges entre le 01/03/2015 et le 31/05/2015 .		AUC, PRC	Augmentation de l'AUC de la courbe précision-rappel de la nouvelle méthode à 18.1% pour les transactions effectuées en face-à-face et de 9,3% pour les transactions d'e-commerce.
[39]	Sparse Autoencoder, GAN .	European dataset.		ScoreF1, Précision, Rappel.	Le GAN entrainé sur les caractéristiques apprises par le Sparse Autoencoder dans la classe majoritaire a amélioré le scoreF1 et la précision, mais avec une diminution du rappel.

TABLE 3.5 – Comparaison des solutions de détection des fraudes sur les cartes bancaires via l'apprentissage semi-supervisé Partie 1

Ref	**Technique**	**Jeu de données**	**Pré traitement**	**Métriques**	**Résultat**
[39]	GAN, Arbre de décision.	European dataset.		Précision, Rappel.	Amélioration de près de 30% de la performance rappel du modèle après le suréchantillonnage.
[39]	GAN, MLP.	European credit card dataset.		Rappel, Spécificité, ScoreF1.	Augmentation de la performance quand le classificateur MLP est entraîné sur le jeu de donnée augmenté.
[39]	WGAN, Régression logistique.	European cardholders credit cards 2013.		ScoreF1, Précision, Rappel.	La régression logistique avec le suréchantillonnage basé sur le WCGAN a produit un scoreF1 et un AUC plus élevés que le GAN.

TABLE 3.6 – Comparaison des solutions de détection des fraudes sur les cartes bancaires via l'apprentissage semi-supervisé Partie 2

3.5 Approches basées sur des méthodes hybrides

Outre les méthodes supervisées, non supervisées, semi-supervisées pour la détection des fraudes, ils existent d'autres approches plus atypiques. Ainsi donc plusieurs chercheurs ont exploré d'autres alternatives crédibles pour la détection des fraudes, le tableau 3.7 présentent une liste d'alternatives crédibles pour la détection des fraudes.

La première solution à laquelle nous allons nous intéresser est celle des auteurs Sharmila Subudhi, Suvasini Panigrahi qui dans leur article paru en 2020 proposent une solution à base de règle floue moyenne (Fuzzy C-Means clustering) et d'un classificateur supervisé pour la détection des fraudes à l'assurance automobile [39]. Bien que cette

solution concerne les fraudes sur les assurances automobile, la méthodologie proposée reste crédible pour la détection des fraudes sur les cartes de crédit. Cette nouvelle approche hybride que proposent les auteurs a été développée pour traiter efficacement le problème du déséquilibre des classes et réduire l'erreur de classification. Fuzzy C-Means (FCM) clustering est une technique de regroupement qui tente de trouver les groupes significatifs présents dans un ensemble de données en leur attribuant des valeurs d'appartenance comprises entre [0 et 1] [39]. Fuzzy C-Means (FCM) clustering a été utilisé avec succès dans différentes applications telles que l'analyse des signaux, la segmentation des images, l'expression des gènes et la détection des fraudes [39]. Le classificateur choisi par les auteurs est le classificateur GMDH (GroupMethodforDataHandling), il s'agit d'un algorithme d'apprentissage supervisé utilisé pour modéliser des systèmes non linéaires complexes. Cet algorithme tente d'établir une relation polynomiale quadratique entre les variables d'entrée et de sortie dans l'ensemble de données d'apprentissage de manière itérative afin de minimiser l'erreur générée lors de la prédiction (différence entre la valeur prédite et la valeur réelle). Les auteurs commencent par extraire un ensemble de données de test du jeu de données d'assurance original déséquilibré. La première étape consiste à appliquer un sous-échantillonnage sur les points de données d'entraînement déséquilibrés en éliminant les valeurs aberrantes présentes dans l'ensemble d'entraînement après avoir appliqué le regroupement avec le Fuzzy C-Mean. On distingue trois groupes de cluster qui sont normaux, suspects ou frauduleux. Les transactions légitimes et frauduleuses sont écartées et les instances suspectes sont analysées individuellement par des classificateurs supervisés pour une classification précise. La GAFCM est une technique de regroupement permettant de réduire le problème de déséquilibre des données. Les échantillons majoritaires normaux de l'ensemble d'apprentissages ont été séparés et le FCM a été entraîné sur ces échantillons à l'aide d'une validation croisée 10 fois. Au total, 4773 instances normales ont été supprimées par le GAFCM à partir de l'ensemble d'entraînement original de 10627 échantillons. L'ensemble de données d'entraînement équilibré a été utilisé pour entraîner deux modèles qui sont le FCM et un GAFCM, qui ont ensuite été comparés à deux modèles FCM et GAFCM identiques entraînés sur le jeu de données d'entraînement déséquilibré. On note une amélioration de la sensibilité, de la spécificité et de l'Accuracy des deux modèles quand ils sont entrainés sur le jeu de donnée équilibré. Le modèle GAFCM a toutefois obtenu les meilleurs résultats, avec une sensibilité de 66,67%, une spécificité de 86,95% et une précision (Accuracy) de 84,34%.

Les méthodes de détection des fraudes basées sur les graphes sont actuellement en hausse en particulier lorsqu'il peut être utile d'analyser les schémas de connectivité dans de grands ensembles de données. Ces techniques se sont avérées particulièrement utiles dans le cadre de la lutte contre le blanchiment d'argent et de la détection des fraudes à l'assurance, où il existe souvent de multiples entités ou organisations pouvant être liées à

la fraude. C'est dans cette logique que les auteurs Lovro Šubelj,Štefan Furlan et Marko Bajec se sont intéressés à l'utilisation des graphes pour la détection des fraudes au niveau des assurances automobiles [39]. Dans le contexte de la fraude des assurances automobiles, plusieurs individus peuvent collaborer pour l'élaboration de la fraude (conducteurs, des chiropracteurs, des garages de réparations, des mécaniciens, des avocats et autres). Il est donc nécessaire dans ce contexte d'observer les relations entre différents individus ou groupes afin d'établir les groupes ou les collaborations pouvant être suspects. Le système proposé par les auteurs est composé de quatre modules utilisés pour la détection des groupes de fraudeurs et des collisions correspondantes. Le premier module se divise naturellement en plusieurs composantes connectées en donnant une représentation du réseau et des différentes connexions existant. Les composants issus du premier module sont analysés par le second module en vue d'identifier les composants considérés comme suspects, pour cela il considère des paramètres comme le diamètre, les cycles, etc. Le troisième module analyse de façon approfondie les composants considérés comme suspects qui ont été produits par le deuxième module, dans l'optique d'identifier les groupes d'entités suspectes et les collisions qui les relient. Le quatrième module est axé sur la visualisation des résultats. Les résultats ont montré que le système proposé était capable de détecter efficacement les cas de fraude et que la représentation appropriée des données était vitale. Ce système présente comme avantage de ne pas avoir besoin de données étiquetées. Le système permet plutôt l'imputation des connaissances d'un expert du domaine, qui peut ainsi être adapté à de nouveaux types de fraude au fur et à mesure qu'ils sont identifiés. L'avantage du système est qu'il ne nécessite pas de grandes quantités de données, les seules difficultés étant qu'il repose sur des seuils ou des paramètres définis par l'utilisateur qui doivent être affinés.

D'autres auteurs se sont intéressés à l'utilisation de solution atypique pour la détection des fraudes, c'est le cas des auteurs Tahmid Hasan, Kazi Tamzid Akhter Md Hasib, Tahsinur Rahman, Akm Bahalul Haque qui dans leurs articles préconise la combinaison de la Blockchain et de l'apprentissage automatique pour une détection efficace des fraudes [60]. La blockchain est une technologie progressive qui a attiré l'attention des milieux de la recherche et des affaires après le succès massif du Bitcoin. La blockchain est une technologie de registre distribué qui stocke les enregistrements de transactions ou de données à l'aide de la cryptographie [60]. Les enregistrements de transactions ou de données sont stockés sous forme de blocs d'informations sur un réseau peer-to-peer [60]. Le tout premier bloc est connu sous le nom de bloc de genèse, et chaque bloc est connecté au bloc précédent [60]. L'essence de la blockchain est de conserver les enregistrements des transactions de manière décentralisée, immuable, transparente, disponible et sécurisée, tout en garantissant l'anonymat [60]. Les solutions basées sur la blockchain établissent des supports hautement sécurisés et dignes de confiance pour les affaires et les échanges commerciaux tels que l'assurance, les

transactions d'investissement, le financement par capital-risque, etc. Les expérimentations ont été effectuées sur le jeu de données publiques "PaySim Synthetic Financial Datasets for Fraud Detection" qui est disponible sur Kaggle [60]. Seul 0.0012% des données sont des données de fraudes ce qui fait environ 8213 transactions sur 6.3 millions. Comme nous pouvons le voir, le jeu de données est fortement déséquilibré. Les auteurs pour leur expérimentation ont considéré les métriques comme l'Accuracy, la Sensibilité (Taux de vrais positifs), Précision, Rappel, Score-FBeta, Taux de faux négatifs. Les auteurs expliquent le choix de ces métriques en raison du fort déséquilibre du jeu de données. La première étape de leurs méthodologies vise à résoudre le déséquilibre présent dans le jeu de données, pour cela les auteurs utilisent SMOTE en suréchantillonnage. Il est important de mentionner que les auteurs ont appliqué SMOTE non pas sur l'ensemble des données, mais sur les données d'entrainement et de tests. Les auteurs par la suite on entrainer cinq modèles d'apprentissage automatique que sont Bernoulli naïf bayes, multinomial naïf bayes, classificateur passif agressif, descente de gradient stochastique et perceptron. Les résultats des expérimentations montrent que c'est l'algorithme classificateur passif agressif qui a la meilleure performance avec un Score-FBeta de 94%, également il présente le plus faible taux de faux négatif qui est 4.1%. L'algorithme le moins performant parmi ceux testés est le Bernoulli naïf bayes avec un ScoreF-Beta de 69% et un taux de faux négatif de 38%. La dernière consiste à déployer le classificateur passif agressif sur le réseau Blockchain. Le flux du processus peut être résumé en trois étapes, la première étape est le déploiement du modèle d'apprentissage automatique sur le système. La seconde étape est le déploiement du contrat intelligent (smart contract) par un régulateur. La troisième étape est l'alimentation en données de l'algorithme d'apprentissage automatique par des acteurs comme les sites d'e-commerce.

Ref	**Technique**	**Jeu de données**	**Pré traitement**	**Métriques**	**Résultat**
[39]	Fuzzy C-Mean(FCM), Genetic Algorithm(GA).	Ensemble de données d'assurance (non spécifié).		Accuracy, Spécificité, Sensibilité.	Augmentation de la performance des deux algorithmes, quand entraîner sur le jeu de données équilibrées.
[39]	Algorithme d'évaluation itérative (IAA) .	Jeu de données non mentionné.		Rappel, Spécificité, ScoreF1.	Le système permet plutôt l'imputation des connaissances d'un expert du domaine, qui peut ainsi être adapté à de nouveaux types de fraude au fur et à mesure qu'ils sont identifiés.
[60]	Blockchain, SMOTE, Bernoulli naïf bayes, multinomial naïf bayes, classificateur passif agressif, descente de gradient stochastique, perceptron	PaySim Synthetic Financial Datasets for Fraud Detection.	√	Accuracy, la Sensibilité, Précision, Rappel, Score-FBeta, Taux de faux négatifs.	l'algorithme classificateur passif agressif qui a la meilleure performance avec un Score-FBeta de 94%, également il présente le plus faible taux de faux négatif qui est 0.041.

TABLE 3.7 – Comparaison des méthodes hybrides de détection des fraudes Partie 1

Chapitre 4

Techniques d'exploration des données

Les techniques d'exploration des données sont le processus d'analyse des données de très grandes tailles dans le but de découvrir des modèles, des tendances et d'avoir une idée de la manière dont ces données peuvent être utilisées [36],[14]. Les techniques d'exploration des données sont l'une des étapes particulières pour découvrir la connaissance dans les données qui peuvent aider à la prise de décisions, cette discipline combine à la fois les statistiques , l'apprentissage automatique, la recherche dans les bases de données pour l'extraction de la connaissance dans les données comme l'illustre les figures 4.1. Les techniques d'exploration des données repose principalement sur trois disciplines qui sont les suivantes [36],[32] :

- La statistique : Sans la statistique les techniques d'exploration des données n'aurait pas vu le jour, en effet la statique amène des techniques qui peuvent résumer l'analyse exploratoire des données et permettre d'identifier les relations systématiques entre différentes variables dans le contexte ou la quantité de données n'est pas suffisante [36]. Nous pouvons mentionner les techniques suivantes :

  Méthodes informatiques : Statistiques descriptives (distributions, paramètres statistiques classiques (moyenne, médiane, écart-type, etc.), corrélation, tableaux de fréquences multiples, techniques exploratoires multivariées (analyse en grappes, analyse factorielle, analyse en composantes principales et analyse de classification, analyse canonique, analyse discriminante, arbres de classification, analyse des correspondances), modèles linéaires/non linéaires avancés (régression linéaire/non linéaire, séries chronologiques/prévisions, etc ;) [36]

  La visualisation des données vise à représenter les informations sous une forme visuelle et peut être considérée comme l'une des méthodes d'exploration

des données les plus puissantes et les plus attrayantes. Parmi les techniques de visualisation les plus courantes, on trouve : Les histogrammes de toutes sortes (colonnes, cylindres, cônes, pyramides, camemberts, barres, etc.), les diagrammes en boîte, les diagrammes de dispersion, les diagrammes de contour, les diagrammes matriciels, les diagrammes d'icônes, etc [36].

- L'intelligence artificielle contribue par la mise en place des modèles basés sur le raisonnement humain qui permette l'exploration des données. l'apprentissage automatique sous discipline de l'intelligence artificielle contribue énormément dans le processus de l'exploration de données [36].
- Les systèmes de base de données fournissent les informations à extraire en appliquant les techniques de statistique et d'intelligence artificielle mentionnées [36].

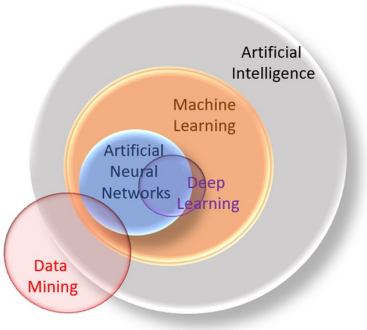


FIGURE 4.1 – Relations entre l'IA, le Machine Learning et les autres techniques d'exploration de données : Source : researchgate

4.1 Prétraitement des données

Les données dans le monde réel peuvent être non arrangées, inconsistantes, incomplètes, incorrectes. Dans ce contexte le prétraitement des données est l'une des étapes importantes permettant d'améliorer la qualité des données qui pourront par la suite être traitées de façon effective dans la phase d'exploration des données [36],[10]. Cette étape a pour but d'éliminer toute anomalie dans les données, cela permet d'améliorer la qualité des données, d'avoir des données consistantes, facilite la lecture, l'utilisation et l'interprétation des algorithmes d'apprentissage automatique. Le prétraitement est composé de plusieurs sous-étapes qui seront décrites par la suite [36],[10].

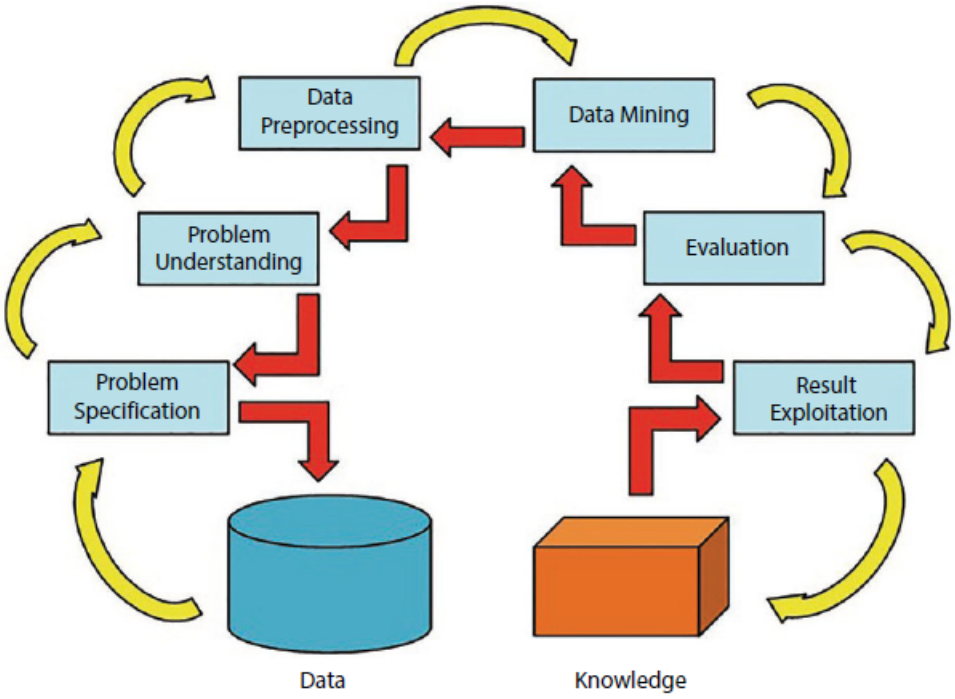


FIGURE 4.2 – Étapes du processus de découverte des connaissances. : Source : Data mining and Machine learning applications

4.1.1 Nettoyage des données

Le nettoyage des données est une étape nécessaire lorsque les données brutes sont inconsistantes , contiennent des erreurs ou sont incomplètes [61]. La duplication des données est prise en charge dans l'étape du nettoyage des données. De façon plus simple, le nettoyage des données peut être décrit comme le processus de fixer ou de supprimer les données incorrectes , corrompues (contenant des erreurs) , dupliquées , incomplètes ou ayant un mauvais formatage à l'intérieur d'un ensemble de données [61]. La mauvaise qualité des données impacte de façon négative la performance des algorithmes d'apprentissage automatique et fausse les résultats d'analyse même si ceux-ci peuvent paraître corrects.

4.1.2 Intégration des données

Durant la phase d'analyse, nous collectons des données provenant de plusieurs sources, ses données doivent être intégrées pour réduire l'inconsistance des données et la redondance, car ces données peuvent exister dans différentes bases de données [61]. De façons plus simples l'intégration des données est le processus consistant à combiner des données provenant de plusieurs sources en des données uniques pour une vue unifier (figure 4.3).

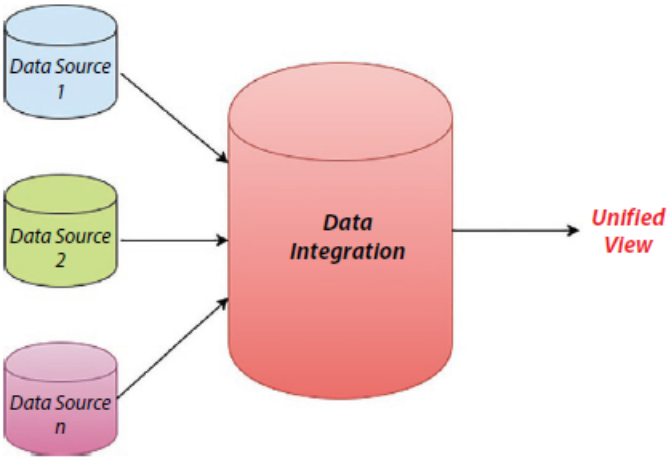


FIGURE 4.3 – Vue unifiée des données : Source : Data mining and Machine learning applications

4.1.3 Réduction des données

La réduction des données consiste à réduire la dimension des données en des données de petite dimension [61],[14]. Plusieurs techniques sont utilisées pour la réduction des données parmi les plus populaires, nous avons les techniques suivantes :

- Agrégation de cubes de données : Dans cette technique, les données sont réduites par l'application d'opérations OLAP telles que le découpage, le dédoublement ou l'enroulement [36].

- Réduction de la dimensionnalité : Les attributs ou dimensions des données sont réduits. Tous les attributs ne sont pas nécessaires pour l'exploration des données. Le sous-ensemble d'attributs le plus approprié est sélectionné à l'aide de techniques telles que la sélection en amont, l'élimination en aval, l'induction par arbre de décision ou une combinaison de sélection en amont et d'élimination en aval [36].

- Compression des données : Cette technique permet de compresser de grands volumes de données, c'est-à-dire de réduire le nombre de bits utilisés pour stocker les données. Cela peut se faire en utilisant la compression avec ou sans perte. Dans la compression avec perte, la qualité des données est compromise pour une plus grande compression. Dans la compression sans perte, la qualité des données n'est pas compromise pour un faible niveau de compression [36].

- Réduction de la numérabilité : Cette technique permet de réduire le volume des données en choisissant des formes plus petites pour la représentation des données. La réduction de la numérotation peut être effectuée à l'aide d'histogrammes, de regroupements ou d'échantillonnages de données. La réduction de la numérabilité

est nécessaire car le traitement de l'ensemble des données est coûteux et prend du temps [36].

4.1.4 Transformation des données

La transformation des données permet de transformer ou de convertir les données d'un format en un format de données acceptables [3],[36]. Les techniques les plus courantes pour la transformation des données sont les suivantes :

- La normalisation est une technique utilisée pour réduire la redondance dans des données et garantir la bonne utilisation du stockage des données. Lorsque les valeurs des attributs d'un tableau ont des échelles différentes. Il est difficile d'analyser les données à l'aide des techniques d'exploration des données, il est donc nécessaire de normaliser les données en fixant les valeurs des attributs entre une plage fixe de 0 a 1 ou de -1 a 1 pour ce faire nous pouvons utiliser différentes techniques de normalisation en fonction du contexte.

- Dans de nombreux cas, nous travaillons avec des ensembles de données dans lesquels nous avons un grand nombre d'attributs , une des techniques permettant de réduire la redondance des données est la sélection des attributs qui permet d'éliminer les attributs ne contenant pas d'informations nécessaires cela rend plus efficace la phase de prétraitement des données.

- La discrétisation des données consiste à diviser de larges collections de données ou de valeurs d'attribut en des intervalles, cela consiste à la transformation de données quantitatives en données qualitative. Cela permet une meilleure évaluation des données. Par exemple, si l'attribut "âgé" contient les valeurs 12, 15, 16, 18, 20, 33, 37, 43, 47, 60, 71, 74, 78, 80 comme l'âge des personnes, nous pouvons discrétiser les valeurs en jeunes, matures et vieux comme suit :

  Jeune : 12, 15, 16, 18, 20
  Mature : 33, 37, 43, 47
  Vieux : 71, 74, 78, 80

4.2 Applications des techniques d'exploration des données

L'avènement des techniques d'exploration des données a conduit à la multiplication des possibilités de business, cette branche de la statistique computationnelle compare des millions de données provenant des entreprises pour détecter et prédire le comportement des clients [36]. Cette capacité prédictive des techniques d'exploration des

données a changé la façon dont les entreprises planifient leurs stratégies de business, avec les techniques d'exploration des données on peut comprendre le présent et anticiper le futur. Elle est utilisée dans plusieurs domaines dans cette section, nous vous présenterons certains domaines clés de son application.

4.2.1 Détection de fraude

Les techniques d'exploration des données permet de découvrir les modèles valides et invalides dans des ensembles de données, il existe plusieurs techniques d'exploration des données permettant la découverte des fraudes(anomalies figure 4.4) dans des ensembles [10],[36],[5]. Nous vous présenterons différentes solutions pour la détection des fraudes(anomalies) dans des ensembles de données.

- **Détection des fraudes à l'aide des règles floues :** Les fraudes dans le système bancaire sont de nature multiple et diverses , cependant les détournements de fonds et la présentation de fausse facture sont les plus courants [19],[52]. Les techniques à base de règle floue peuvent dans ce contexte identifier les modèles de transaction jugés frauduleux et prendre les actions qui s'avèrent nécessaires. Plusieurs anomalies ont été répertoriées dont les plus fréquentes sont les suivantes :
  - Les employés qui partageaient le même nom de famille et le même numéro de compte bancaire.
  - Des employés ayant des noms de famille différents et partageant le même numéro de compte bancaire.
  - Les employés dont les numéros d'identification n'étaient pas valides, par exemple des numéros d'identification non numériques ou un nombre incorrect de chiffres.
  - Les salariés ayant le même nom de famille et la même date de naissance.
  - Des employés sur la liste de paie qui ont dépassé l'âge de la retraite obligatoire de 65 ans.
  - Vendeurs ayant des noms différents et partageant une même adresse.
  - Les vendeurs qui partagent le même compte bancaire et le même nom.
  - Les cas où un employé et un fournisseur partagent un compte bancaire.
  - 1000 factures aberrantes pour les 1000 premiers fournisseurs.
  - Les factures en double pour le même fournisseur.

- **Détection des fraudes à l'aide de l'apprentissage automatique et des techniques d'exploration des données :** l'apprentissage automatique fait référence aux techniques analytiques qui permettent de découvrir un ou des modèles dans un ensemble de données sans l'intervention d'un analyste humain ou d'un expert [5],[46],[67],[2]. l'apprentissage automatique permet de détecter les

transactions qui sont susceptibles d'être des fraudes en permettant la reconnaissance de modèle dans de grands ensembles de données. Il existe plusieurs techniques d'apprentissage automatique, mais dans le contexte de la détection de fraude, les plus utilisées sont l'apprentissage supervisé et non supervisé.

La détection de fraude via l'apprentissage supervisé vise a étiqueter un grand ensemble de données financier comme des données frauduleuses ou non frauduleuses qui seront par la suite utilisées pour entrainer,valider et tester notre modèle. Il est également important de souligner que la précision du modèle dépend fortement de la qualité et de la véracité des données étiquetées. Les algorithmes d'apprentissage supervisés les plus utilisés sont les suivants : SVM, les arbres de décision, les réseaux de neurones.

L'apprentissage non supervisé intervient dans le contexte où les données étiquetées sont limitées ou inexistantes. Cette technique fonctionne par auto-apprentissage pour découvrir le ou les modèles dans l'ensemble de données. Les algorithmes d'apprentissage non supervisés les plus utilisés sont les suivants : K-means , SVD clustering.

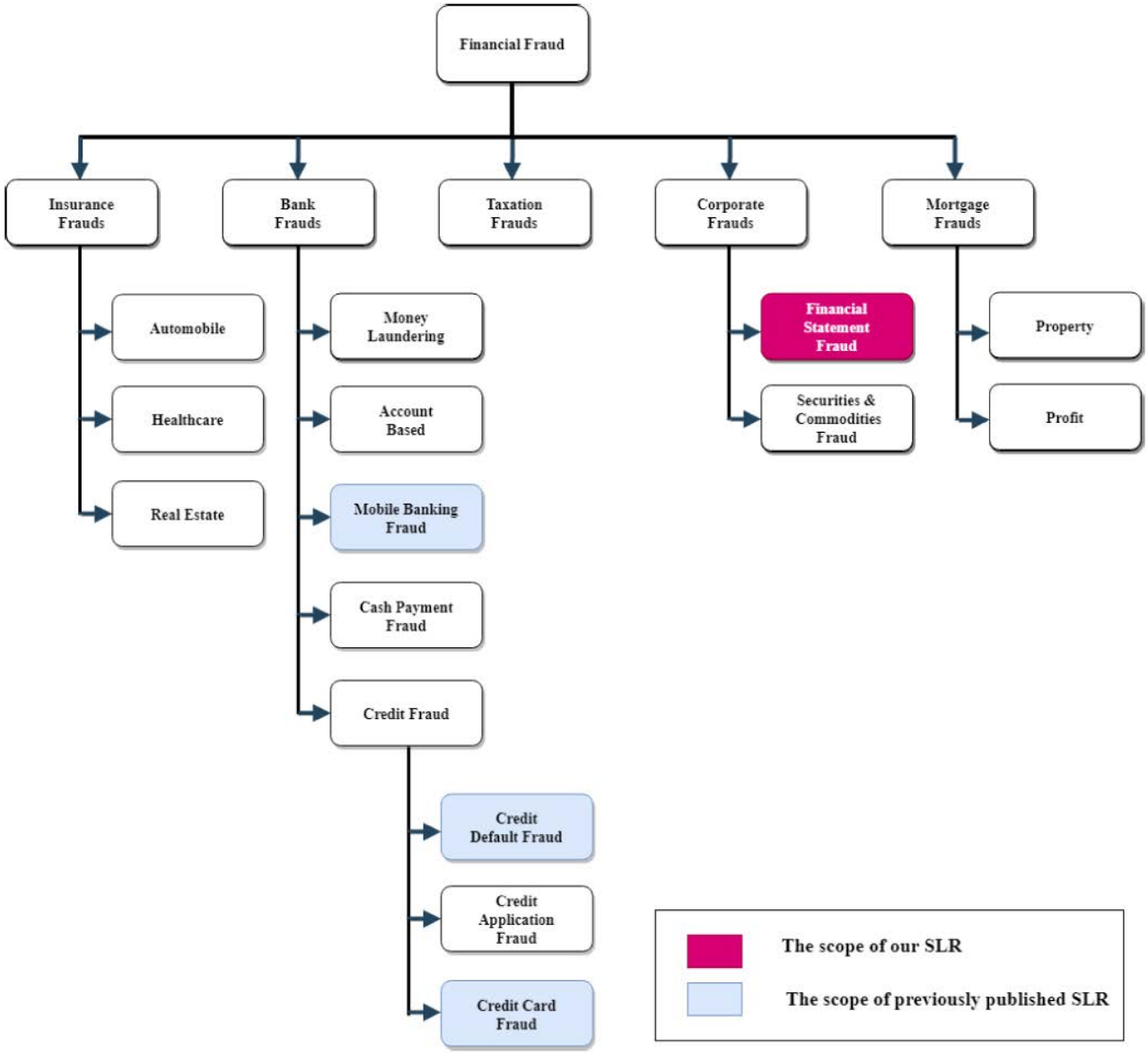


FIGURE 4.4 – Fraudes financières : Source : Intelligent Fraud Detection in Financial Statements Using Machine Learning and Data Mining: A Systematic Literature Review

4.2.2 Autres domaines d'application des techniques d'exploration des données

En dehors de la détection de fraude, les techniques d'exploration des données sont utilisés dans une multitude de domaines allant du domaine de la santé à celui de la sécurité, de l'analyse des marchés financiers, de la vente au détail etc [36],[10],[23],[5]. Les techniques d'exploration des données doivent traiter de large flux de données, il est donc logique que les algorithmes puissent traiter les données sans compromettre les performances du système. Voici ci-dessous les tableaux 4.1, 4.2, 4.3 présentant de façons brèves les domaines d'application des techniques d'exploration des données et pour chaque domaine on présente un cas d'utilisation.

Domaines	**Cas d'utilisation**
Communications	Les méthodes d'exploration des données peuvent être utilisées comme outil pour segmenter les clients en fonction de leurs cibles et de leurs intérêts.
Assurance	Il peut être utilisé comme outil dans le secteur de l'assurance pour identifier les gains de l'entreprise, analyser les politiques existantes et l'intérêt des clients pour ces politiques.
Éducation	Il peut être utilisé comme un outil de suivi pour améliorer le processus d'apprentissage de l'enseignement par les étudiants, et également pour aider les étudiants à choisir le cours de manière efficace.
Production	Il peut être utilisé dans l'industrie opérationnelle et manufacturière pour identifier les équipements défectueux. Il peut analyser et gérer leurs ressources à l'aide d'outils d'exploration de données.
Banque	En utilisant le modèle des techniques d'exploration des données, les banques peuvent analyser le comportement financier des clients. Grâce à ces données, les professionnels des banques peuvent identifier les clients défaillants et les clients fidèles.

TABLE 4.1 – Domaines d'application et cas d'utilisation des techniques d'exploration des données partie 1

Domaines	**Cas d'utilisation**
Vente au détail	L'exploration de données est utile pour le secteur de la vente au détail car elle permet de collecter un grand nombre de données sur les clients (ventes,historique des achats,consommation).
Fournisseurs de services administratifs	La plupart des entreprises de télécommunication basées sur les services peuvent l'utiliser comme outil d'analyse. Il peut être utilisé pour identifier les raisons pour lesquelles les clients quittent leurs services et pour vérifier les griefs des clients. Il peut ainsi améliorer leurs services du point de vue de l'utilisateur.
E-Commerce	La plupart des sites de commerce électronique utilisent des outils d'exploration de données pour attirer les clients en leur proposant différents plan stratégiques.

TABLE 4.2 – Domaines d'application et cas d'utilisation des techniques d'exploration des données partie 2

Domaines	**Cas d'utilisation**
Super marchés	Les supermarchés peuvent observer leurs clients quotidiens en fonction de leur sexe, de leur âge et de leur préférence. Par exemple, supposons que la plupart des clients féminins visitent le supermarché, ils peuvent alors se concentrer sur les produits comme les crèmes pour le visage, les shampooings, les produits pour bébés, les produits sanitaires, etc.
Enquêtes criminelles	Grâce aux techniques d'exploration des données, les enquêteurs peuvent suivre les personnes qui tentent de franchir les frontières et les données peuvent être envoyées aux officiers de police locaux.
Bioinformatique	Il peut également être utilisé dans le domaine scientifique et biomédical.

TABLE 4.3 – Domaines d'application et cas d'utilisation des techniques d'exploration des données partie 3

4.3 Algorithmes d'exploration de données

Les algorithmes des techniques d'exploration des données sont principalement utilisés pour la modélisation prédictive, cette modélisation peut être regroupée en deux groupes qui sont les problèmes de regroupement et de classification que nous vous présenterons par la suite [36],[61].

- Dans le domaine des techniques d'exploration des données, la classification vise à modéliser les données et à les répartir en classes. C'est un processus dans lequel un ensemble de données est réparti en deux ou plusieurs catégories. Considérons la problématique consistant à déterminer si un groupe d'individus est atteint du diabète ou pas ce problème peut être considéré comme un problème de classification binaire avec deux classes qui sont diabétiques ou non-diabétiques. Les principaux algorithmes des techniques d'exploration des données pour les problèmes de classification sont les suivants :

  Régression logistique

  Naïve Bayes

  K plus proches voisins

L'arbre de décision

Random forest

Modèle vectoriel de support(SVM)

- Le clustering consiste à regrouper des groupes d'objets en se basant sur leurs similarités, cette technique repose sur l'utilisation d'algorithmes non supervisés qui font le regroupement des données en se basant sur un certain nombre de critères propres aux données. Cette technique est utilisée dans divers secteurs :

  Analyse du panier de biens et services

  Reconnaissance des formes

  Traitement d'image

  Analyse financière

Chapitre 5

Contributions

5.1 Introduction

Ce mémoire traite des problématiques se rapportant aux détections des fraudes sur les cartes de crédit via l'utilisation des algorithmes d'apprentissage automatique. Le processus de détection de fraude de notre méthode repose sur l'utilisation d'algorithmes évolués. Notre méthode vise à la fois l'amélioration de la performance des modèles de détection de fraude, mais également la robustesse des modèles de détection de fraudes contre les attaques alimentées par l'intelligence artificielle qui sont de plus en plus utilisées par les fraudeurs.

Ce chapitre peut être subdivisé en deux grandes parties. La première partie est axée sur la présentation de la méthodologie ainsi que de l'architecture de notre solution. Dans la seconde partie, nous présenterons les résultats obtenus lors de notre phase d'expérimentation.

5.2 Description de notre approche

5.2.1 Architecture globale de notre système

Dans l'optique de résoudre les déséquilibres présents dans nos jeux de données, nous avons exploré plusieurs techniques afin de déterminer les techniques les plus performantes et appropriées dans notre contexte. La première technique que nous avons explorée est celle du suréchantillonnage, qui fut l'une des premières méthodes proposées dans la résolution du déséquilibre des données [84]. Cette technique repose sur la duplication des éléments de la classe minoritaire, cela n'apporte aucune information nouvelle dans les données. Cette technique présente comme problème majeur le risque d' "surajustement" des données contenu que les données dupliquées ne fournissent aucune information supplémentaire sur les classes sous-représentées [84]. Toujours dans cette logique, nous nous sommes intéressés à la technique de SMOTE de suréchantillonnage

qui génère des données synthétiques en effectuant une interpolation linéaire entre les échantillons de la classe minoritaire et leurs voisins les plus proches [22],[12],[31]. L'un des points forts de cette technique c'est qu'elle ajoute davantage d'informations dans l'ensemble de données, ce qui permet d'améliorer les performances du modèle. Cependant, SMOTE présente comme inconvénient le risque de pouvoir introduire du bruit parmi les instances générées. ADASYN est une technique de suréchantillonnage alternative qui vise à résoudre le problème de la génération d'échantillons synthétiques dans des régions de l'espace des caractéristiques qui sont plus proches de la limite de décision [84]. ADASYN a comme inconvénient majeur d'augmenter la complexité informatique due à la génération d'échantillons synthétiques, ce qui peut affecter le temps d'apprentissage des modèles d'apprentissage automatique. Outre cela, les données générées peuvent ne pas représenter avec précision la véritable distribution sous-jacente de la classe minoritaire [84]. Le GAN est également une autre alternative permettant de générer des données de haute qualité. Ils ne nécessitent pas de données étiquetées pour l'entraînement, ce qui les rend particulièrement utiles dans les scénarios où les données étiquetées sont rares ou coûteuses à obtenir [22],[35],[54],[53]. L'un des problèmes du GAN est son instabilité durant sa phase d'entraînement à cause du Processus d'entraînement adverse. Le GAN nécessite généralement d'importantes ressources informatiques pour l'apprentissage, notamment des GPU très performants et de grandes quantités de mémoire. Le processus d'apprentissage peut être long et coûteux en termes de calcul.Notre choix s'est porté sur SMOTE et GAN en raison de leurs performances.

L'analyse de la figure 5.1 qui présente de façon détaillée l'architecture de notre système, nous permet d'identifier 6 grandes phases. Dans la première phase, nous allons nous intéresser à la résolution du déséquilibre présent dans nos données en utilisant SMOTE et GAN.

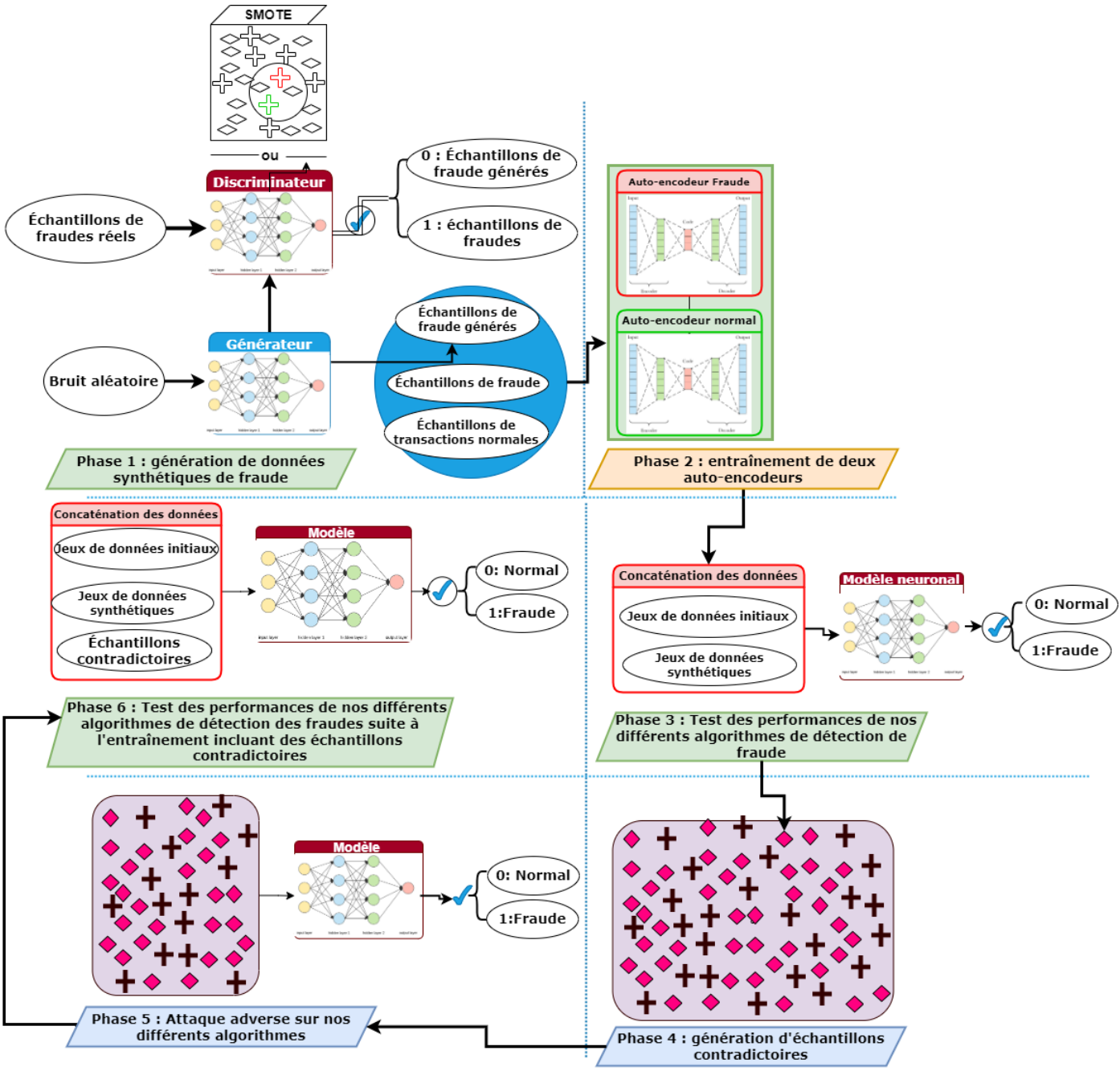


FIGURE 5.1 – Architecture de notre système

Après avoir fini la phase 1 de notre solution, nous allons obtenir un jeu de données qui est équilibré cependant il est encore possible d'améliorer notre solution. C'est cela qui nous a conduits dans la phase 2 à nous intéresser à l'utilisation des Autoencodeurs pour augmenter davantage la diversité de notre jeu de données. Contrairement aux applications classiques de l'Autoencodeurs pour la détection des fraudes par le calcul de l'erreur de reconstruction, dans notre contexte nous utiliserons deux Autoencodeurs comme générateurs de données synthétiques [27],[7]. Ce choix s'explique par la capacité des Autoencodeurs à apprendre les caractéristiques essentielles présentes dans nos données.

La phase 3 est réservée aux entraînements et aux tests de performance de nos cinq algorithmes d'apprentissage automatique. Ces cinq algorithmes ont été choisis en fonction de leurs performances dans les détections des fraudes sur les cartes de crédit. À cette étape, nous allons concaténer notre jeu de données qui est composé de nos données initiales ainsi que ceux synthétiques, ces données augmentées seront scindées en deux parties (entraînement et test) et passer à chacun de nos cinq modèles d'apprentissage comme données d'entrée. Les résultats de nos tests seront amplement détaillés dans la suite de notre document.

Dans la phase 4, nous allons générer un ensemble d'échantillons contradictoires grâce à la bibliothèque **"Adversarial Robustness Toolbox"**, cette étape sera amplement détaillée dans la suite de notre travail [21],[49]. Ces échantillons contradictoires seront fortement utiles dans la phase 5 du processus de notre solution.

La phase 5 vise à tester la robustesse de nos différents modèles de détection des fraudes, pour cela nous utiliserons les échantillons contradictoires qui ont été générés dans la phase précédente. Les échantillons contradictoires ont pour but de tromper notre modèle en l'induisant à classifier des échantillons de fraude comme étant des échantillons normaux. [13],[21].

Dans la phase 6, nous allons améliorer la robustesse de notre modèle en incluant les échantillons contradictoires durant la phase d'entraînement et de test de nos différents algorithmes [13],[21]. Les résultats obtenus seront amplement analysés et discutés dans la suite de notre travail.

5.2.2 Présentation des jeux de données

L'un des défis majeurs que nous rencontrons dans la mise en place d'un système pour la détection des fraudes demeure la difficulté d'accès aux données qui sont pour la plupart du temps confidentiel [39]. Les transactions financières contenant des données de nature très sensible, les organisations financières sont très peu enclin au partage de ces données même pour des buts de recherche de solutions pour l'identification des fraudes [39]. Comme nous le savons, les algorithmes d'apprentissage automatique requièrent de très grandes quantités de données pour leur apprentissage, ce qui rend crucial l'accès aux données. Dans notre recherche de jeux de données, deux jeux de données ont particulièrement retenu notre attention en raison de leur pertinence et qualité, outre cela ces deux jeux de données sont constamment utilisés dans la plupart des articles traitant de la détection des fraudes sur les cartes de crédit [45] [68].

Notre premier jeu de données contient des transactions effectuées par cartes de crédit en septembre 2013 par des détenteurs de cartes européens [45]. Ces transactions ont eu

lieu en deux jours où nous avons 492 fraudes sur 284 807 transactions. L'ensemble de données est fortement déséquilibré, la classe positive (fraudes) représentant 0,172% de toutes les transactions. Il ne contient que des variables d'entrée numériques qui sont le résultat d'une transformation par l'analyse en composante principale. Cette transformation a été effectuée pour des raisons de confidentialité. Les caractéristiques ont donc été renommées V1, V2, ... V28 , les seules caractéristiques qui n'ont pas été transformées avec l'ACP sont les caractéristiques "Temps", "Montant" et "Classe". Ce jeu de données totalise un ensemble de 31 caractéristiques (colonnes) dans laquelle la caractéristique "Temps" contient les secondes écoulées entre chaque transaction et la première transaction de l'ensemble de données. La caractéristique "Classe" est la variable de réponse et prend la valeur 1 en cas de fraude et 0 dans le cas contraire.

Notre deuxième jeu de données est issu d'une simulation de transactions par carte de crédit contenant des transactions légitimes et frauduleuses pour la période allant du 1er janvier 2019 au 31 décembre 2020 [68]. Ces données ont été générées à l'aide de l'outil "Sparkov Data Generation" simulant des transactions effectuées sur une période d'une année. Contrairement à notre premier jeu de donnée , ici aucune transformation n'a été effectuée. Ce jeu de donnée est composé de 1 296 675 transactions dont 7506 sont des transactions frauduleuses. À l'image de notre premier jeu de donnée, ce jeu de donnée est fortement déséquilibré. On note un total de 22 caractéristiques (colonnes) dont certaines colonnes contiennent des valeurs non numériques.

5.2.3 Prétraitement des deux jeux de données

Avant toute utilisation de nos données, il est important d'effectuer en premiers lieux le prétraitement des données pour les rendre plus adaptés à nos algorithmes d'apprentissage automatique [61],[36]. Le prétraitement est une étape de l'exploration des données et de l'analyse des données qui a pour but de transformer des données brutes en données pouvant être comprises et analyser par les ordinateurs et les algorithmes d'apprentissage automatique [61],[36]. Cette étape permet l'élimination de toute anomalie dans l'ensemble de données. Cette étape fait référence à la manipulation, au filtrage ou à l'augmentation des données avant leur analyse. Le prétraitement des données est une étape importante du processus d'exploration des données permettant la résolution des valeurs aberrantes, manquantes, redondantes ou tout autre problème que présentent nos données brutes. Il peut inclure des phases comme le nettoyage des données, la normalisation des données, l'encodage des données, la transformation des données, l'extraction et la sélection des caractéristiques dans nos données. Cette étape vise, entre autres, à transformer les données dans un format pouvant être plus facilement et efficacement traité par les algorithmes d'apprentissage automatique. Il permet également l'obtention de résultats plus précis avec les algorithmes d'apprentissage automatique. Dans notre contexte, les principales étapes de

prétraitement que nous avons appliquées sont les suivantes :

- La première étape consiste a explorer nos données. Pour cela nous allons importer notre ensemble de données et effectuer une visualisation de nos données de façons claires. Cette étape a pour but de nous permettre de comprendre notre ensemble de données et de voir comment cela peut répondre à nos besoins en effectuant des filtrages, en segmentant nos données, en vérifiant la présence de données dupliquées.

- Recherche de valeurs manquantes : Cette étape consiste pour nous à nous assurer que nous n'avons pas de valeur manquante, dans le cas où nous avons des valeurs manquantes plusieurs techniques et algorithmes peuvent être appliqués pour résoudre ce problème.

- Convertir les variables catégoriques en variables numériques dans notre deuxième jeu de données [68].

- Suppression des colonnes non nécessaires qui sont présentes dans notre deuxième jeu de données [68].

- La caractéristique "Temps" contient les secondes écoulées entre chaque transaction et la première transaction de l'ensemble de données. Cette caractéristique va être convertie en heure de la journée, cela nous permettra d'avoir une meilleure interprétation visuelle de nos données [45].

- Normalisation des données : Cette étape permet la mise à l'échelle de notre ensemble de données sur une plage régularisée, ce qui va nous permettre de pouvoir faire des comparaisons plus précise [36]. Egalement, cela permet d'améliorer la performance et la stabilité du modèle lors de la phase d'entraînement [36]. Dans notre contexte la technique de normalisation choisie est l'échelle minimax qui consiste à mettre nos données sur une plage fixe qui est comprise entre 0 et 1. MinMax repose sur la théorie du jeu, dans laquelle est la valeur la plus élevée que le joueur peut être sûr d'obtenir sans connaître les actions des autres joueurs, et Min est la valeur la plus basse que les autres joueurs peuvent forcer le joueur à recevoir lorsqu'ils connaissent l'action du joueur. Elle est définie par l'équation $v_i = max_{a_i} min_{a-i} v_i(a_i, a_{-i})$ [81]. Dans cette équation i est l'indice de l'acteur concerné, -i désigne tous les autres joueurs à l'exception du joueur i, a_i est l'action entreprise par le joueur i, a_{-i} représente les actions entreprises par tous les autres joueurs et v_i est la fonction de valeur du joueur i.

5.2.4 Génération de données synthétique avec l'algorithme SMOTE

La grande majorité des techniques d'apprentissage automatique présentent de faibles performances sur les ensembles de données déséquilibrées. Cela s'explique par le fait que ces techniques d'apprentissage ignorent les classes minoritaires alors que ce sont généralement ces classes qui sont les plus importantes [1],[43].

L'une des approches qui par le passé étaient utilisées pour traiter les ensembles de données déséquilibrés consistait simplement à dupliquer les exemples de la classe minoritaire, cependant en dupliquant la classe minoritaire nous n'ajoutons aucune information nouvelle à notre ensemble de données. Une autre solution qui donne de meilleurs résultats consiste à synthétiser de nouvelles données à partir d'exemples de données existants. Cette technique est connue sur le terme anglais SMOTE qui fait référence au suréchantillonnage des minorités synthétiques [22],[12],[31]. Il s'agit d'une technique de statistique qui permet d'augmenter le nombre d'échantillons d'une classe donnée de sorte à rendre notre ensemble de données plus balancé. Les nouvelles données générées par SMOTE ne sont pas des copies des données excitantes, ils combinent les caractéristiques du cas cible avec celles de ses voisins. Dans notre contexte nous utiliserons SMOTE pour générer des données synthétique de fraudes, suite a cela nous procéderons a une comparaison avec l'algorithme de réseau génératif adverse dans le but de déterminer lequel des deux algorithmes génère de meilleures données synthétiques qui sera utilisé dans la suite de notre méthodologie.

5.2.5 Entrainement des réseaux de neurones adverse

Le réseau de neurones génératifs adverses a pour but de permette la génération de données similaires aux données de fraudes réelles, dans cette logique les données de fraudes générées suivront la même distribution que les données de fraudes réelles présents dans notre jeu de données [35],[54],[53]. Les données de fraude générées par le réseau de neurones adverse seront par la suite combinées aux données d'entrainement pour former des données d'entrainement augmentées, cette technique est plus connue sur le terme d'augmentation de données. Cela permet de résoudre la problématique des données déséquilibrées de notre jeu de données en augmentant la quantité de données représentant des transactions de fraudes.

L'entrainement d'un réseau génératif adverse s'avère difficile, car si le générateur est nettement plus performant que le discriminateur l'ensemble de notre réseau ne sera pas bien entrainé et inversement si le discriminateur est peut performant alors les fausses données générées tromperont facilement le discriminateur et le générateur ne s'améliorera pas lors du prochain cycle d'apprentissage. Les deux composantes

(discriminateur et générateur) de notre réseau sont en concurrence pour l'emporter sur l'autre, ils sont donc fortement dépendants l'un de l'autre pour un entrainement efficace.

Les données d'entrainement x_f seront utilisées par la suite pour entrainer l'un des autoencodeurs responsables de la représentation des données de fraudes. Cela permettra à l'autoencodeur d'apprendre une bonne représentation des données de fraudes et d'éviter une sous-représentation causée par un nombre insuffisant d'échantillons de fraude dans notre ensemble de données. Notre architecture est composée de trois couches cachées, qui prennent en entrer un bruit d'une dimension 30 pour le premier jeu de données et 22 pour le second jeu de données à partir duquel il génère des données synthétiques de fraudes. Le générateur est muni d'une couche de sortie d'une dimension correspondant à la dimension de chacun des jeux de données. Le discriminateur de notre architecture prend en sortie un neurone muni d'une activation sigmoïde. La première couche cache est composée de 100 neurones, la deuxième couche cache est composée de 200 neurones et la troisième couche cache est composée de 400 neurones. Nous notons également dans notre architecture la présence de couches d'optimisation. Le GAN est entraîné pendant 5000 itérations, et nous avons utilisé comme optimisation la fonction Adam à cause de sa forte convergence. Le taux d'apprentissage le plus optimal dans notre contexte est $1e^{-5}$ et la taille du lot est de 128.

5.2.6 Entrainement des Autoencodeurs doubles

Dans notre contexte, nous utiliserons les Autoencodeurs pour la génération de données synthétiques qui seront par la suite combinés à nos jeux de données pour augmenter la quantité et la diversité des données sur lesquelles les différents modèles seront entraînés. Dans cette logique, nous entraînerons deux Autoencodeurs l'un responsable de la génération de données synthétiques de fraude à partir des données de fraude générées par notre réseau génératif adverse, le second Autoencodeur quant a lui sera responsable de la génération de données synthétiques normales [79],[27].

Les deux Autoencodeurs seront entraînés sur nos deux jeux de données, cela permettra d'améliorer la performance des différents modèles en augmentant la quantité et la diversité des données d'entraînement, de test, ainsi que de validation des différents modèles. Bien que notre réseau génératif adverse nous génère des données synthétiques de fraudes, nous avons choisi l'Autoencodeur également comme technique de génération à cause de sa capacité à apprendre les caractéristiques essentielles présentes dans nos données, cela permettra d'augmenter la diversité présente dans nos données [79],[27]. Notre Auto-encodeur est composé de trois couches cachées. Au niveau de l'encodeur, la première couche est composée de 32 neurones, la seconde couche est composée de 64 neurones et la troisième couche 128 neurones. Le décodeur, quant à lui, a une première couche cachée de 182 neurones, la deuxième couche est de 64 neurones, et la troisième

couche est de 32 neurones. Nous notons également la présence de couche d'optimisation de type Relu, l'optimisateur choisi est Adam avec un taux d'apprentissage de $1e^{-5}$. L'autoencodeur est entraîné par validation croisée durant 500 itérations.

5.2.7 Entraînement et comparaison des performances des différents modèles de classification

Les algorithmes de classification que sont XGBoost, Random Forest, LightGBM, SVM, et la régression linéaire sont ceux que nous utiliserons pour l'entraînement et les tests de performance. Notre choix s'est porté sur ces algorithmes du fait de leurs performances dans la résolution des problématiques se rapportant aux fraudes sur les cartes bancaires [5],[56],[10].

Ces différents algorithmes de classification seront entraînés et testés sur nos deux jeux de données [45], [68]. Dans un premier temps nous entraînerons nos modèles sur nos deux ensembles de données initiales qui à la base sont fortement déséquilibrés (asymétriques), par la suite nous procéderons à une comparaison de la performance des différents modèles. Dans la seconde étape, chaque modèle sera entraîné par cross-validation et testé sur nos deux jeux de données augmentés. Les jeux de données augmentés sont composés des données initiales, auxquelles nous avons ajouté les données de fraude synthétique générées par notre réseau génératif adverse ainsi que ceux de générés par les deux Autoencodeurs (données synthétiques de fraude et normales). En entraînant, nos modèles sur les deux jeux de données augmentés, nous pourrons améliorer drastiquement la performance des différents modèles, également nous déterminerons parmi les différents algorithmes que nous avons utilisés lequel s'avère le plus performant pour la détection de fraudes.

5.2.8 Amélioration de la robustesse des différents modèles

La dernière étape de notre travail consistera à essayer d'améliorer la robustesse des différents modèles d'apprentissage automatique que nous avons utilisés pour la détection des fraudes sur les cartes bancaires. Dans cette logique, nous utiliserons la bibliothèque connue sur le terme anglais de **"Adversarial Robustness Toolbox"** qui est une bibliothèque python permettant aux développeurs et aux chercheurs d'évaluer, de défendre, de certifier et de vérifier les modèles et les applications d'apprentissage automatique contre les menaces adverses que sont l'évasion, l'empoisonnement, l'extraction et l'inférence.

Cette bibliothèque a également comme avantage de supporter la majorité des frameworks populaires (TensorFlow, Keras, PyTorch, MXNet, scikit-learn, XGBoost, LightGBM, CatBoost, GPy, etc.), en plus de cela elle supporte différents types de formats (images, tableaux, audio, vidéo, etc.) et peut être utilisée pour des tâches de

classification, détection d'objets, de génération et de certification. Toutes ces raisons ont motivé notre choix pour cette bibliothèque.

Nous allons implémenter une attaque ZOO (Zeroth Order Optimization) avec la bibliothèque **"Adversarial Robustness Toolbox"** pour tester la performance de nos différents modèles avec les paramètres de confidence de 0.5, un taux d'apprentissage de $1e^{-3}$, le nombre d'itérations maximum est de 200, une étape de recherche binaire de 10 et une taille du lot de 128. Nous avons choisi ZOO en raison de son efficacité, et de ça s'impliciter d'implémentation, outre cela, ZOO est inclus par défaut dans la bibliothèque **"Adversarial Robustness Toolbox"**. ZOO (Zeroth Order Optimization) est une attaque de type boîte noire qui nécessite l'accès aux données d'entrée et de sortie (indice de confiance) d'un modèle [49]. Il s'agit d'un algorithme d'optimisation fonctionnant selon une tendance itérative, partant d'un point donné x^0, il génère une séquence d'itérations x^k (ou solutions d'essai) qui convergent vers une solution. Mathématiquement parlant, le scalaire x^k converge vers 0 si et seulement si, pour tous les nombres réels $\varepsilon > 0$, il existe un entier positif K tel que $|x^k| < \varepsilon$ pour tout k$\geq K$. Alors X^k converge vers une solution X^* si et seulement si $||X^k - X^*||$ converge vers 0.

L'idée principale de l'attaque ZOO est d'approximer les gradients du modèle cible à l'aide de méthodes d'optimisation d'ordre zéro (sans dérivation), ce qui permet à l'attaquant de générer des exemples contradictoires sans accès direct aux gradients ou à l'architecture du modèle [13]. Les étapes de l'attaque ZOO sont les suivantes :

- ❖ Approximer les gradients à l'aide d'une optimisation d'ordre zéro, telle que la méthode de différence finie coordonnée ou la méthode basée sur les coordonnées sphériques.
- ❖ Calculer la perturbation contradictoire à l'aide des gradients approximatifs.
- ❖ Appliquer la perturbation sur les données originales en s'assurant que les exemples contradictoires demeurent dans la même plage que nos données originales.

Les impacts de cette attaque sur nos différents modèles feront l'objet d'analyses. Les résultats des analyses de cette attaque nous permettront de proposer des solutions efficaces pour contrer les attaques adverses qui sont par nature très subtiles et difficiles à contrer. Parmi les solutions existantes pour contrer ce type d'attaque, celle qui a retenu notre attention est l'entraînement contradictoire plus connu sur le terme anglais de **"Adversarial training"**. Nous avons choisi cette solution à cause de son efficacité, de sa performance et de sa simplicité à être implémenté [21].

L'entraînement adverse est une méthode utilisée pour améliorer la robustesse d'un

modèle en incluant lors de la phase d'entraînement du modèle, les échantillons contradictoires ayant réussi à tromper le modèle [21]. Ainsi donc nous allons réentrainer les différents modèles en incluant les échantillons adverses et par la suite, nous procéderons à de nouveaux tests de performance des différents modèles.

5.3 Environnement logiciel

5.3.1 Présentation du langage python

Python est un langage de programmation de haut niveau interprété, orienté objet et doté d'une sémantique dynamique. Ses structures de données intégrées de haut niveau, combinées au typage dynamique et à la liaison dynamique, le rendent très intéressant pour le développement rapide d'applications, ainsi que pour une utilisation en tant que langage de script ou de collage pour relier des composants existants. Python est un langage simple, facile d'apprentissage avec une syntaxe intuitive, il supporte également plusieurs modules et packages ce qui encourage une programmation modulaire et la réutilisation du code. Le langage Python a été développé par Guido van Rossum au début des années 1980 et a été publier pour la première fois le 20 février 1991. Python tire son nom d'une ancienne série de sketches de la BBC intitulée Monty Python's Flying Circus (Le cirque volant des Monty Python), c'est un langage qui permet d'augmenter la productivité dans le sens ou il n'y a pas d'étape de compilation ce qui rend le débogage incroyablement rapide lorsque l'interpréteur détecte une erreur une exception est alors générer.

5.3.2 Présentation de la plateforme Anaconda

Anaconda est une plateforme open source qui permet d'écrire et d'exécuter des codes dans le langage de programmation Python. Le but d'Anaconda est de simplifier le déploiement et la gestion des packages Python. Les versions des packages d'Anaconda sont gérées par le système de gestion de package Conda qui analyse l'environnement actuel avant l'installation de tout package afin d'éviter tout conflit entre différents packages. La distribution d'Anaconda a environ plus de 250 packages qui sont installés automatiquement en plus de cela on a environ plus de 7500 packages qui peuvent être installés. Anaconda en plus d'avoir une interface de ligne de commande possède une interface graphique qui permet de lancer des applications de contrôler les packages Conda et les environnements d'exécution sans faire recours à la ligne de commande.

5.3.3 Présentation de Visual Studio Code

Visual Studio Code est un éditeur de code source disposant d'outils puissants comme la complétion de code, le débogage. Il est disponible sur Mac Os, Linux et Windows, en plus cela il dispose d'un éditeur rapide avec un support pour plus d'une centaine de langages de programmation dont le langage python.Visual studio Code permet d'être plus

productif grâce à l'indentation automatique, la coloration syntaxique, la refonte de code, etc, en plus de cela il possède plusieurs extensions installables dont l'une le permet de se connecter à la distribution python installer sur votre ordinateur il peut donc utiliser l'interpréteur Python du gestionnaire de package Anaconda. Visual Studio Code dispose d'outils de construction du script, il est également compatible avec Git ce qui vous permet de travailler avec le contrôle de source sans quitter l'éditeur, y compris la visualisation des différences de changements en cours.

5.3.4 Présentation de la bibliothèque tensorflow

Tensorflow est une bibliothèque logicielle libre et gratuite utilisée pour l'apprentissage automatique et l'intelligence artificielle. Il peut être utilisé pour plusieurs taches, mais il est essentiellement utilisé pour l'entrainement et l'inférence des réseaux de neurones profonds. Tensorflow a été développé par Google Brain(équipe de recherche en intelligence artificielle), la première version a été publiée sous la licence Appache 2.0 en 2015. Tensorflow est compatible avec une multitude de langages de programmation incluant Python, Javascript, C++ et Java ce qui permet d'être utilisable dans différents secteurs.

Tensorflow peut s'exécuter sur plusieurs CPU et GPU(CUDA, SYCL), il est également disponible sur plusieurs systèmes d'exploitation comme Linux, Mac Os, Windows, Android, IOS. Son architecture flexible de déployer facilement des calculs sur une variété de plateformes(CPU, GPU, TPU) des ordinateurs de bureau aux clusters de serveurs en passant par les appareils mobiles et périphériques. Le nom Tensorflow provient des opérations que ces réseaux neuronaux effectuent sur des tableaux de données multidimensionnelles, appelés tenseurs.

En mai 2016 Google à annoncer son unité de traitement de tenseurs, un circuit intégré spécifique à une application (ASIC, une puce matérielle) construit spécifiquement pour l'apprentissage automatique et adapté à Tensor Flow. Une TPU est un accélérateur d'IA programmable conçu pour fournir un débit élevé d'arithmétique de faible précision (par exemple, 8 bits) et orienté vers l'utilisation ou l'exécution de modèles plutôt que leur entrainement.

Pour résumer, nous pouvons dire que Tensorflow et une plateforme libre puissante pour l'apprentissage automatiquent, il se concentre sur le développement et l'entrainement des modèles d'apprentissage automatique. Il est capable d'entrainer plusieurs catégories de réseau de neurones comme les réseaux de neurones convolutif(reconnaissance d'image), les réseaux de neurones récurrents, séquentiels, le traitement de langue naturelle et bien d'autres réseaux de neurones.

5.3.5 Présentation de la bibliothèque Keras

Keras est une bibliothèque logicielle libre qui fournit une interface Python pour les réseaux neuronaux artificiels. Keras supporte plusieurs backends comme Tensorflow, Microsoft Cognitive toolkit, Theano et Plaid Ml. Il a été conçu pour permettre une expérimentation rapide sur les réseaux de neurones profonds. a a été développé dans le cadre des efforts de recherche du projet ONEIROS (Open-ended Neuro-Electronic Intelligent robot operating system) et son principal auteur et mainteneur est François Chollet, un ingénieur de Google.

Keras contient plusieurs implémentations communes sur les réseaux de neurones comme les couches de réseaux de neurones, les fonctions d'activation(sigmoid, tangente, relu, leaky-relu.....) , des fonctions d'optimisation ainsi qu'une série d'outils permettant de travailler plus facilement avec des données d'images et de texte afin de simplifier le codage nécessaire à l'écriture du code des réseaux neuronaux profonds. En plus des réseaux de neurones standard , Keras supporte également les réseaux de neurones convolutionnel et récurrent.

5.3.6 Présentation de la bibliothèque matplotlib

Matplotlib est une bibliothèque permettant de créer des visualisations statiques, animées et interactives en Python. Il fournit une API orientée objet pour l'intégration de graphique dans des applications utilisant des outils d'interface graphique générale tels que Tkinter, wxpython, Qt ou GTK. Matplotlib a été écrit à l'origine par John d. Hunter. Depuis lors, il a bénéficié d'une communauté de développement actif et est distribué sous une licence du type BSD.

Pyplot est un module de Matplotlib qui fournit une interface similaire à MATLAB. Matplotlib est conçu pour être aussi utilisable que MATLAB, avec la possibilité d'utiliser Python, et l'avantage d'être gratuit et open source. Plusieurs boîtes à outils sont disponibles pour étendre les fonctionnalités de Matplotlib. Certains sont téléchargeables séparément, d'autres sont livrés avec le Code source de Matplotlib. les bibliothèques de Matplotlib sont les suivantes :

- Cartopy : Une bibliothèque cartographique comprenant des définitions de projection cartographique orientées objet et des capacités de transformation arbitraire de points, de lignes, de polygones et d'images.

- Outils Excel : Utilitaires pour l'échange de données avec Microsoft Excel3.

- Outils GTK : Interface avec la bibliothèque GT.

- ❖ Interface Qt.
- ❖ Mplot3d : Tracés en 3D.
- ❖ Natgrid : Interface avec la bibliothèque natgrid pour le maillage de données irrégulièrement espacées.
- ❖ Seaborn : Fournit une API au-dessus de Matplotlib qui offre des choix raisonnables pour le style de tracé et les couleurs par défaut, définit des fonctions simples de haut niveau pour les types de tracés statistiques courants, et s'intègre avec les fonctionnalités fournies par Pandas.

5.3.7 Présentation de la bibliothèque Pandas

Panda est une bibliothèque du langage de programmation python destiné à la manipulation et à l'analyse des données. Il est spécialisé pour la manipulation des tableaux numériques et des séries chronologiques publiées sous la licence BSD à trois clauses. Le nom est dérivé du terme "données de panel", un terme d'économétrie pour les ensembles de données qui comprennent des observations sur plusieurs périodes de temps pour les mêmes individus. Panda a été créer par Wes McKinney un développeur de logiciels libres spécialisé dans l'informatique analytique entre les années 2007 à 2010 périodes pendant laquelle il faisait de la recherche.

Pandas est principalement utilisé pour l'analyse de données et la manipulation associée de données tabulaires dans des DataFrames. Pandas permet d'importer des données à partir de différents formats de fichiers tels que les valeurs séparées par des virgules, JSON, Parquet, les tables ou les requêtes de bases de données SQL et Microsoft Excel. Pandas permet diverses opérations de manipulation de données telles que la fusion, le remodelage, la sélection, ainsi que le nettoyage des données et les fonctions de manipulation des données. Le développement de pandas a permis d'introduire dans Python de nombreuses fonctionnalités comparables à celles du langage de programmation R pour travailler avec des DataFrames. La bibliothèque pandas est construite sur une autre bibliothèque, NumPy, qui est orientée vers le travail efficace avec des tableaux plutôt que sur les caractéristiques du travail avec des DataFrames.

5.3.8 Présentation de la bibliothèque sklearn

Scikit-learn également connue sous le nom de sklearn est une bibliothèque gratuite d'apprentissage automatique du langage python. Il propose divers algorithmes de

classification, de régression et de regroupement, notamment les machines à vecteurs de support, les forêts aléatoires, le gradient boosting, les k-means et DBSCAN, et est conçu pour interopérer avec les bibliothèques numériques et scientifiques NumPy de Python.

Le projet scikit-learn a vu le jour sous la forme de scikits.learn, un projet Google Summer of Code réalisé par le scientifique français David Cournapeau. Le code de base originelle a ensuite été réécrit par d'autres développeurs. En 2010, les contributeurs Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort et Vincent Michel, de l'Institut français de recherche en informatique et en automatique de Saclay, en France, ont pris la direction du projet et ont publié la première version publique de la bibliothèque le 1er février 2010.

scikit-learn est largement écrit en Python et utilise NumPy de manière intensive pour l'algèbre linéaire haute performance et les opérations sur les tableaux. En outre, certains algorithmes de base sont écrits en Cython pour améliorer les performances. Les machines à vecteurs de support sont implémentées par un wrapper Cython autour de LIBSVM , la régression logistique et les machines à vecteurs de support linéaires par un wrapper similaire.scikit-learn s'intègre bien avec de nombreuses autres bibliothèques Python, telles que Matplotlib et plotly pour le traçage, NumPy pour la vectorisation des tableaux, Pandas dataframes, SciPy, et bien d'autres encore.

5.3.9 Présentation de la bibliothèque xgboost

XGBoost, qui signifie Extreme Gradient Boosting (renforcement du gradient extrême) est une bibliothèque d'apprentissage automatique distribuée et évolutive d'arbres de décision boostés par le gradient (GBDT). Elle permet de booster les arbres en parallèle et constitue la principale bibliothèque d'apprentissage automatique pour les problèmes de régression, de classification et de classement. Il est également important de mentionner qu'elle est disponible pour plusieurs langages de programmation incluant Java,C++,Python , R , Julia , Scalla et supporter sur par plusieurs systèmes d'exploitation comme Linux , Windows , MacOS.Il fonctionne sur une seule machine, ainsi que sur les frameworks de traitement distribué Apache Hadoop, Apache Spark, Apache Flink et Dask.

XGBoost a commencé comme un projet de recherche par Tianqi Chen dans le cadre du groupe Distributed (Deep) Machine Learning Community (DMLC). Initialement, il s'agissait d'une application de terminal pouvant être configurée à l'aide d'un fichier de configuration libsvm. Le modèle XGBoost permet souvent d'obtenir une précision supérieure à celle d'un seul arbre décisionnel, mais il sacrifie l'interprétabilité intrinsèque des arbres décisionnels. Par exemple, suivre le chemin emprunté par un arbre de décision pour prendre sa décision est trivial et s'explique de lui-même, mais suivre les chemins de

centaines ou de milliers d'arbres est beaucoup plus difficile.

5.4 Expérimentations et résultats

5.4.1 Analyse exploratoire des données

Avant toute implémentation de notre solution, l'exploration de nos deux jeux de données s'impose [45] , [68]. En explorant notre premier jeu de données (figure 5.4), nous nous sommes rendu compte que ce jeu de données est composé de deux classes qui ont les labels 0 pour les transactions normales et 1 pour les transactions de fraudes [45]. Au total , ce jeu de données est composé de 284807 transactions parmi lesquelles 0.172% représentent des transactions de fraudes, soit un total de 492 transactions de fraudes [45]. Une exploration plus approfondie de ce jeu de données nous révèle que nous avons un total de 31 colonnes, à l'exception des colonnes temps, montant, et classes les 28 autres colonnes on subit l'algorithme d'analyse en composante principale pour des raisons de confidentialité et son numéroté de V1 a V28 [45]. À cause de l'application de l'analyse en composante principale, les colonnes sont faiblement corrélées les unes aux autres, nous pouvons voir cela sur la figure 5.2. On note également l'absence de données incohérente ou nulle dans ce jeu de données , cela nous simplifie la phase de prétraitement des données qui viendra par la suite. Comme mentionné plus haut , ce jeu de données est fortement déséquilibré en ce sens que les transactions représentant des fraudes sont ultraminoritaires. Les colonnes "Classe","Montant","V28" ont particulièrement retenu notre attention, car elles sont fortement asymétriques , on peut voire cela en observant la figure 5.5.

Notre deuxième jeu de données à l'image du premier jeu de données est fortement déséquilibré comme nous pouvons le voir sur les figures 5.6 et 5.7 présentant respectivement le déséquilibre groupé par classe et selon le genre (homme et femme) [68]. Ce jeu de données est composé 1 296 675 transactions dont 1 289 169 sont des transactions normales et 7 506 des transactions de fraude. Aucune colonne nulle , ou contenant des valeurs aberrantes n'est à signaler. Ce jeu de données contient des valeurs catégoriques que nous devrons par la suite convertir en valeurs numériques. Ce choix de conversion des valeurs s'explique par le fait que les valeurs catégorielles ne peuvent généralement pas être traitées directement par les algorithmes d'apprentissage automatique, car la plupart des algorithmes sont principalement conçus pour fonctionner avec des données numériques uniquement. Par conséquent, avant que nos algorithmes puissent utiliser ces valeurs comme entrées, elles doivent être codées sous forme de valeurs numériques(Label Encoding). Ce jeu de données n'a subi aucune transformation également, aucune valeur dupliquée n'est à signaler. Il est composé de 22 colonnes, l'analyse des colonnes montre la présence de colonnes contenant peu d'informations utiles comme les colonnes "Nom","Prénom","Numéro de transaction".

L'analyse de la figure 5.3 présentant la matrice de corrélation nous permet de dire que les colonnes sont faiblement corrélées les unes aux autres. Cependant on note de forte corrélation entre les colonnes "long" et "zip" , également on note une faible corrélation entre la colonne "amt" représentant le montant et la colonne "isfraud". L'analyse de la figure 5.8 présentant la répartition des fraudes par catégorie pour notre deuxième jeu de données nous montre une très légère hausse des fraudes pour les catégories "shopping-net" et "grocery-pos", cela nous suggère que les fraudes touchent essentiellement ces deux catégories. En analysant la figure 5.9 présentant la répartition des transactions selon l'heure, nous constatons un très léger pic pour les heures 22h et 23 ,cela nous suggère que les fraudes ont essentiellement lieu entre 22h et 23h, également nous constatons un très léger pic des fraudes pour les heures 0h à 3h. La synthèse de l'analyse de la figure 5.9 nous suggère que les fraudes ont généralement lieu très tard dans la nuit (22h et 23h) ou très tôt le matin (0h à 3h).

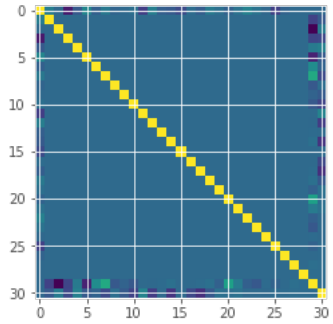


FIGURE 5.2 – Matrice de corrélation du premier jeu de données

	cc_num	amt	zip	lat	long	city_pop	unix_time	merch_lat	merch_long	is_fraud
cc_num	1.00	0.00	0.04	-0.06	-0.05	-0.01	0.00	-0.06	-0.05	-0.00
amt	0.00	1.00	0.00	-0.00	-0.00	0.00	-0.00	-0.00	-0.00	0.21
zip	0.04	0.00	1.00	-0.11	-0.91	0.08	0.00	-0.11	-0.91	-0.00
lat	-0.06	-0.00	-0.11	1.00	-0.01	-0.15	0.00	0.99	-0.01	0.00
long	-0.05	-0.00	-0.91	-0.01	1.00	-0.05	-0.00	-0.01	1.00	0.00
city_pop	-0.01	0.00	0.08	-0.15	-0.05	1.00	-0.00	-0.15	-0.05	0.00
unix_time	0.00	-0.00	0.00	0.00	-0.00	-0.00	1.00	0.00	-0.00	-0.01
merch_lat	-0.06	-0.00	-0.11	0.99	-0.01	-0.15	0.00	1.00	-0.01	0.00
merch_long	-0.05	-0.00	-0.91	-0.01	1.00	-0.05	-0.00	-0.01	1.00	0.00
is_fraud	-0.00	0.21	-0.00	0.00	0.00	0.00	-0.01	0.00	0.00	1.00

FIGURE 5.3 – Matrice de corrélation du deuxième jeu de données

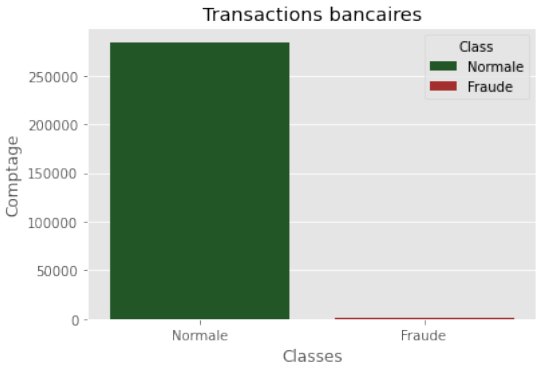
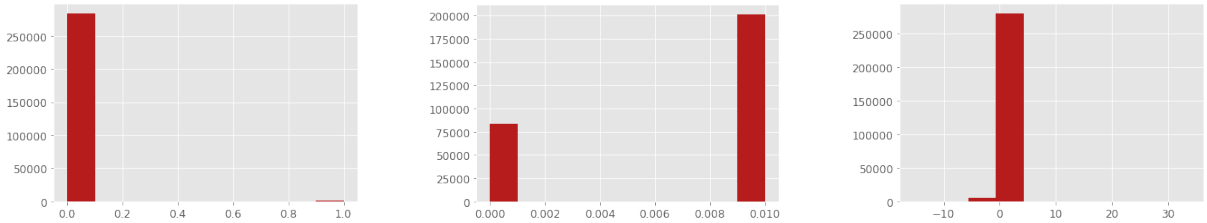


FIGURE 5.4 – Transactions groupées par classe du premier jeu de donnée



(a) Classe (b) Montant (c) Colonne V28

FIGURE 5.5 – Colonnes fortement asymétriques du premier jeu de donnée

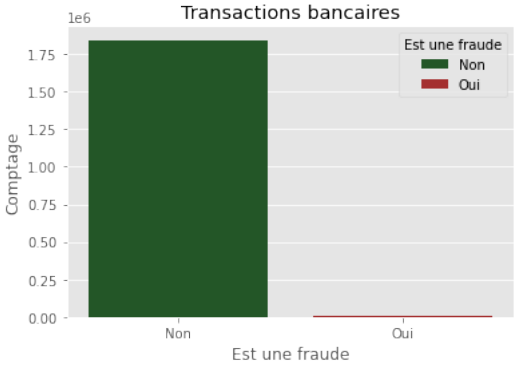


FIGURE 5.6 – Transactions groupées par classe du deuxième jeu de données

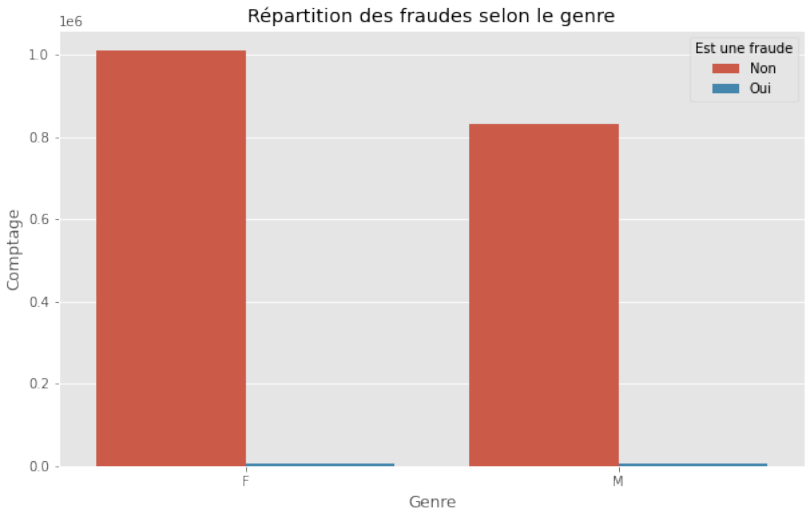


FIGURE 5.7 – Transactions groupées selon le genre pour le deuxième jeu de données

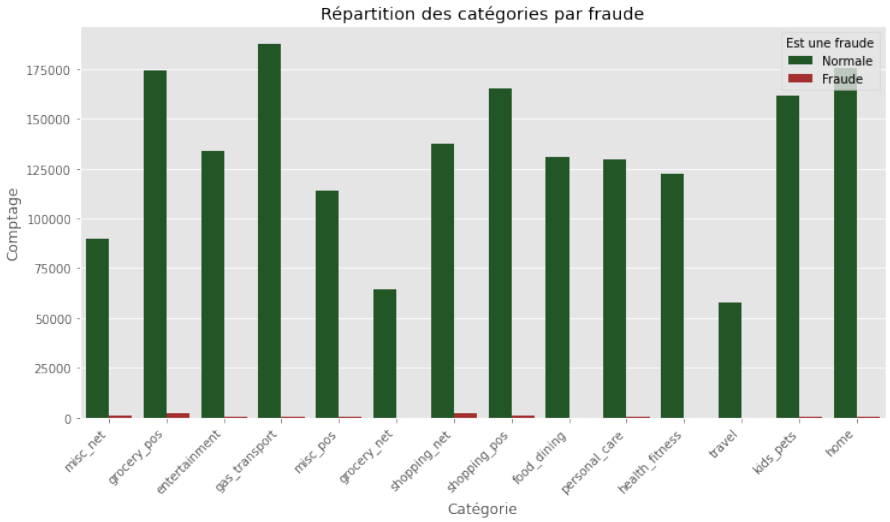


FIGURE 5.8 – Répartition des catégories par fraude pour le deuxième jeu de données

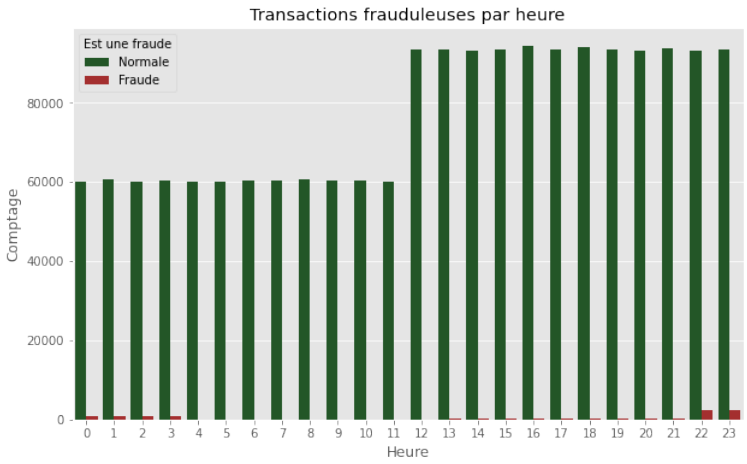


FIGURE 5.9 – Répartition horaire des transactions frauduleuses pour le deuxième jeu de données

5.4.2 Prétraitement des données

Dans la phase de prétraitement, nous procéderons au prétraitement de nos deux jeux de données de façon séparée en fonction de la spécificité de chaque jeu de données. Les différentes des étapes de la phase de prétraitement du premier jeu de données sont les suivantes [45] :

- L'une des premières étapes a été d'appliqué une transformation logarithmique sur la colonne montant afin d'avoir une distribution plus normale comme on peut le voir sur la figure 5.10.

- Convertir la colonne temps qui représente le nombre de secondes qui se sont écoulées entre une transaction et la première transaction de l'ensemble de données en heure de la journée comme l'illustre la figure 5.11. Cette transformation de la colonne temps nous permet d'avoir une meilleure compréhension des périodes de fraude, en effet en analysant la figure 5.11 nous constatons un pic de fraude entre l'intervalle 0 à 5 heures et entre l'intervalle de 10 à 12 heures.

- Dans l'optique de résoudre l'asymétrie des valeurs des colonnes nous avons opté pour une mise à l'échelle des données, pour aboutir a cet objectif nous avons utiliser la fonction MinMaxScaler de la bibliothèque sklearn qui nous a permis de transformer chaque caractéristique (colonne) a l'exception de la caractéristique (colonne) classe en les mettants sur une plage de valeur comprise entre l'intervale [0,1].

Les différentes étapes de la phase de prétraitement du deuxième jeu de données sont les suivantes [68] :

- Suppression des colonnes non essentielles.

- Convertir les valeurs catégoriques des colonnes catégorie et genre en valeur numérique.

- Utiliser la fonction MinMaxScaler de la bibliothèque sklearn qui nous a permis de transformer chaque caractéristique (colonne) à l'exception de la caractéristique (colonne) "isfraud" en les mettant sur une plage de valeur comprise entre l'intervalle [0,1].

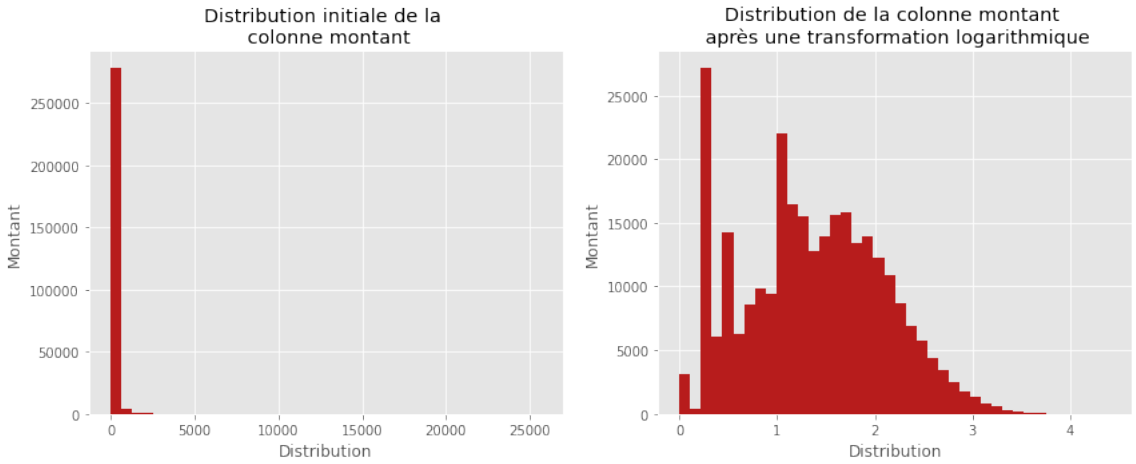


FIGURE 5.10 – Transformation des valeurs de la colonne *montant*

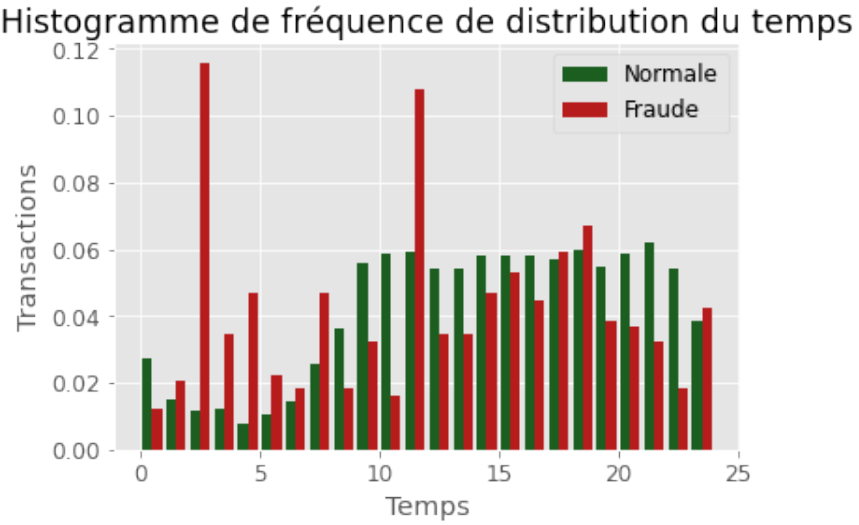


FIGURE 5.11 – Histogramme de fréquence de distribution du temps

5.4.3 Entraînement et test de la performance des différents modèles sur nos ensembles de données déséquilibrées

Dans cette section, nous allons nous intéresser aux performances de nos cinq algorithmes qui vont être testés sur nos deux jeux de données déséquilibrés. Les algorithmes choisis pour effectuer ces différents tests sont XGBoost , RandomForest, LightGBM, SVM et la Régression logistique. Ces algorithmes ont été choisis en raison de leur bonne performance dans la résolution des problèmes de classification et plus particulièrement la détection des fraudes [5],[56],[10]. Les différentes expérimentations effectuées vont nous permettre de comprendre comment chaque algorithme performe sur des jeux de données fortement déséquilibrés. Pour effectuer nos différents tests, nous avons considéré les métriques de performance comme le "Rappel" ,"Précision", "RocAuc", "ScoreF1", "Accuracy", ces métriques vont nous permettre de connaître la performance réelle de nos différents algorithmes. Les résultats des expérimentations sur nos deux jeux

de données sont contenus dans les tableaux 5.1 et 5.2 que nous allons analyser par la suite.

En analysant de plus près le tableau 5.1 et la figure 5.13 qui présente les résultats des performances de nos cinq algorithmes sur notre premier jeu de données , le premier constat qui nous vient a l'œil c'est le pourcentage élevé de la métrique "Accuracy" pour tous nos cinq algorithmes avec des pourcentages respectifs de 99.95% pour l'algorithme XGBoost , 99.95% pour le RandomForest, 99% pour le LightGBM, 99.56% pour le SVM et de 99.23% pour la Régression logistique. L'accuracy représente la mesure globale des prédictions de notre modèle, elle mesure le rapport entre les instances correctement prédites (vrais positifs et vrais négatifs) et le nombre total d'instances. Le pourcentage élevé de la métrique "Accuracy" sur notre jeu de données déséquilibrées peut nous conduire à croire que nos modèles sont très performants alors que ce n'est pas le cas, il nous faut analyser les autres métriques. Le pourcentage élevé de la métrique "Accuracy" peut s'expliquer par le déséquilibre de nos données, car les transactions frauduleuses sont ultraminoritaire en comparaison des transactions normales, cela conduit donc le modèle a prédire uniquement la classe majoritaire [1],[43],[56]. Le faible pourcentage de la métrique "Rappel" par rapport à la métrique "Accuracy" nous indique que nos différents modèles ont un nombre élevé de faux négatifs, ce qui est le résultat du déséquilibre de nos données, en effet les transactions frauduleuses dans notre premier jeu représentent uniquement 0.172% de l'ensemble de nos transactions [1]. L'algorithme LightGBM et Régression logistique présentent les plus faibles pourcentages de "Rappel", XGBoost et SVM sont ceux ayant le plus fort pourcentage de "Rappel" avec un score de 77.02% suivi par RandomForest avec 75%. En analysant le "ScoreF1" qui nous donne la moyenne de la précision et du rappel, nous constatons que XGBoost et RandomForest présentent les plus forts pourcentages avec 84.75% et 84.71%, LightGBM présente le score le plus faible avec 31.36%, ce score s'explique par le faible pourcentage de son "Rappel" et de sa "Précision". Il indique également la faible performance du modèle [1]. Globalement sur notre premier jeu de données XGBoost est l'algorithme le plus performant. Ce résultat peut s'expliquer par sa capacité à pouvoir traiter les données déséquilibrées [1]. En deuxième position nous avons RandomForest, il est suivi par SVM qui arrive en troisième position. La Régression logistique est l'algorithme qui performe le moins bien parmi nos cinq algorithmes que nous avons testés, il est suivi par LightGBM. Ces deux modèles sont biaisés en faveur de la classe majoritaire [1].

L'analyse du tableau 5.2 et de la figure 5.15 montrant les résultats de nos cinq algorithmes sur notre second jeu de données nous amène à la même conclusion que le tableau précédent, cependant pour les algorithmes SVM et régression logistique nous notons un très faible pourcentage du rappel, du ScoreF1 et de la précision qui est de l'ordre de 0%. Ces faibles pourcentages montrent que ces algorithmes sont incapables

d'identifier les transactions frauduleuses parmi le nombre total de transactions. Ces résultats peuvent s'expliquer par la faible performance de ces algorithmes sur des données très déséquilibrées, cela est particulièrement vrai dans notre deuxième jeu de données, dans lequel sur 1 296 675 transactions, seules 7506 sont frauduleuses, soit environ 0,00578% de transactions frauduleuses. Globalement sur notre deuxième jeu de données, XGBoost est l'algorithme le plus performant, il est suivi par RandomForest quanta LightGBM, il arrive en troisième position. La Régression logistique et SVM sont ceux qui performent les moins bien en étant incapables d'identifier les transactions frauduleuses dans notre jeu de données, cela nous suggère que ces deux algorithmes performent très faiblement sur des données très déséquilibrées.

Une synthèse de l'analyse des tableaux 5.1 et 5.2 nous montre que nos algorithmes performent mieux sur notre premier jeu de données que sur le deuxième. Ce résultat peut s'expliquer par le fait que le second jeu de données est plus déséquilibré avec 0,00578% étant des transactions frauduleuses. Outre cela, les analyses des deux tableaux montrent que nos modèles présentent un pourcentage de rappel plus faible en comparaison de la métrique "Accuracy", cela traduit un pourcentage élevé de faux négatifs. Les conclusions de nos analyses des tableaux 5.1 et 5.2 nous montrent que le fort déséquilibre de nos deux jeux impacte fortement la performance des nos cinq modèles , cela se traduit par un pourcentage élevé de faux négatifs qui montre que les modèles ont du mal a identifiées les transactions frauduleuses des transactions normales [1],[43],[56]. La prochaine étape consiste a rééquilibrer nos données dans l'optique d'améliorer la performance de nos différents modèles.

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	77.02%	94.21%	88.50%	84.75%	99.95%
RandomForest	75%	96.52%	87.49%	84.41%	99.95%
LightGBM	57.43%	21.57%	78.53%	31.36%	99%
SVM	77.02%	78.62%	88.49%	77.81%	99.23%
Régression logistique	50%	83.14%	74.99%	62.44%	99.89%

TABLE 5.1 – Performance des différents algorithmes sur le premier jeu de données

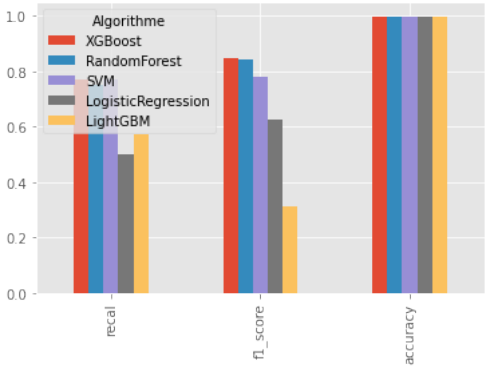


FIGURE 5.12 – Le graphe de performance des différents algorithmes sur le premier jeu de données

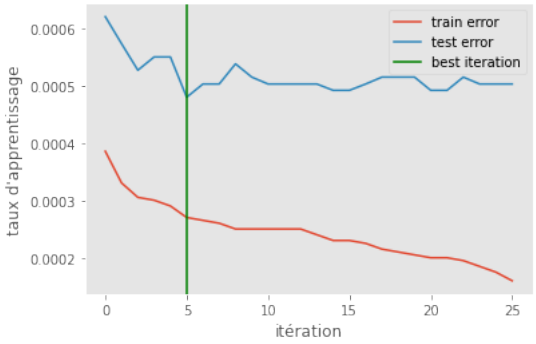


FIGURE 5.13 – Courbes d'apprentissage xgboost sur le premier jeu de données

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	70.98%	85.91%	85.46%	77.73%	99.78%
RandomForest	67.18%	83.44%	83.55%	74.43%	99.75%
LightGBM	57%	51%	78%	54	99%
SVM	0%	0%	50%	0%	99.47%
Régression logistique	0%	0%	49.99%	0%	99.46%

TABLE 5.2 – Performance des différents algorithmes sur le deuxième jeu de données

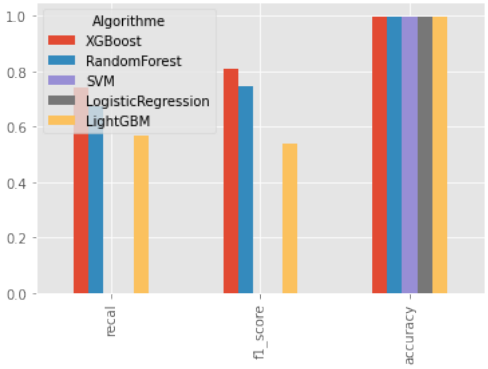


FIGURE 5.14 – Le graphe de performance des différents algorithmes sur le deuxième jeu de données

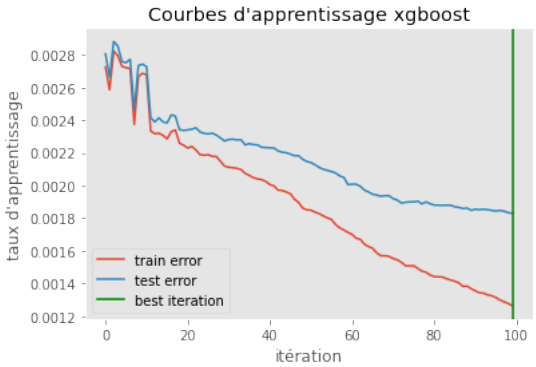


FIGURE 5.15 – Courbes d'apprentissage xgboost sur le deuxième jeu de données

5.4.4 Génération des données synthétiques de fraude avec SMOTE

Dans l'optique de rééquilibrer nos deux jeux de données, nous avons opté pour l'utilisation de l'algorithme SMOTE afin de générer de nouvelles données synthétiques de fraude qui suive la même distribution que nos données de fraude originale comme l'illustre les figures 5.17 et 5.18. SMOTE est une technique de génération de données ayant fait ses preuves dans le rééquilibrage des données déséquilibrées [22],[12],[31]. Comme nous pouvons le voir sur la figure 5.16 après l'application de SMOTE, notre jeu de données est devenu équilibré avec à la fois autant de données normales que de données de fraude.

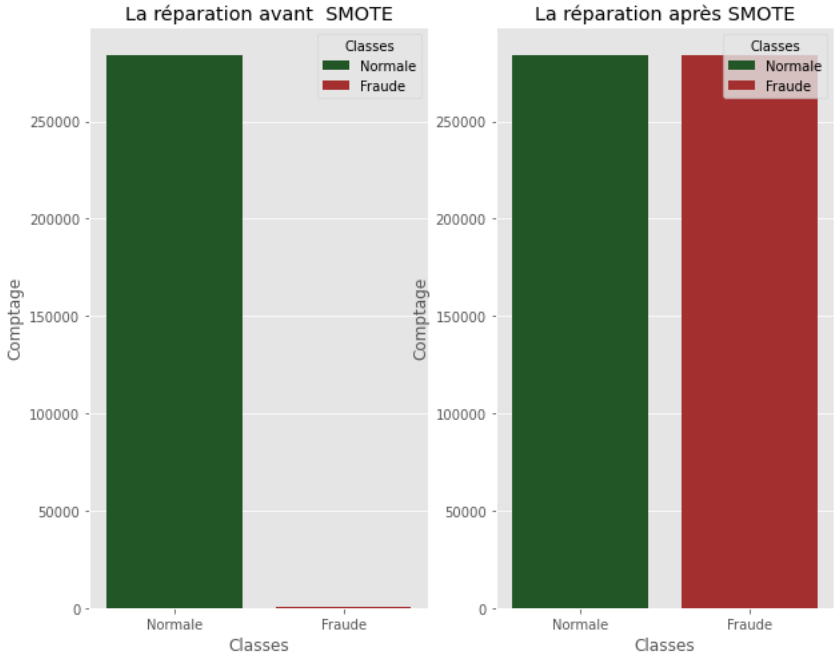


FIGURE 5.16 – Application de l'algorithme SMOTE

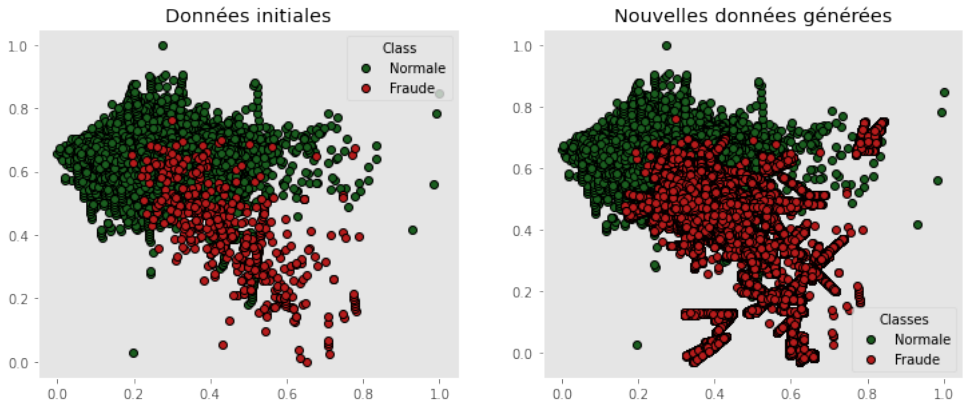


FIGURE 5.17 – Application de l'algorithme SMOTE(jeu de donnée 1)

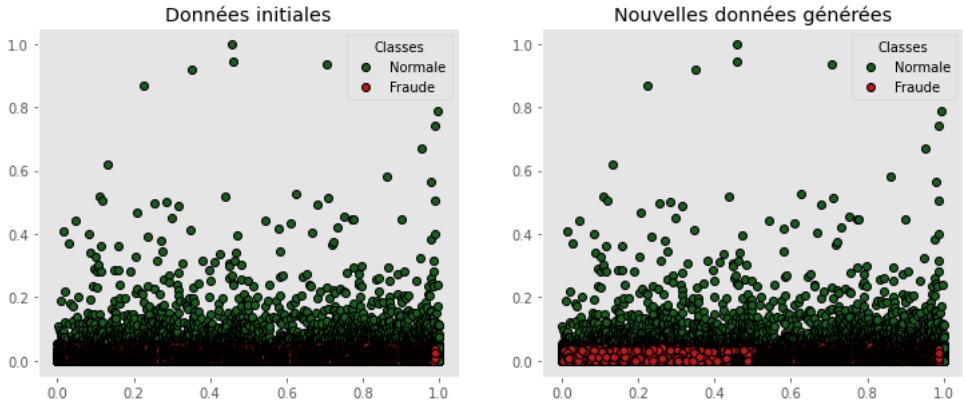


FIGURE 5.18 – Application de l'algorithme SMOTE(jeu de donnée 2)

5.4.5 Génération des données synthétiques de fraude via les réseaux génératifs adverses

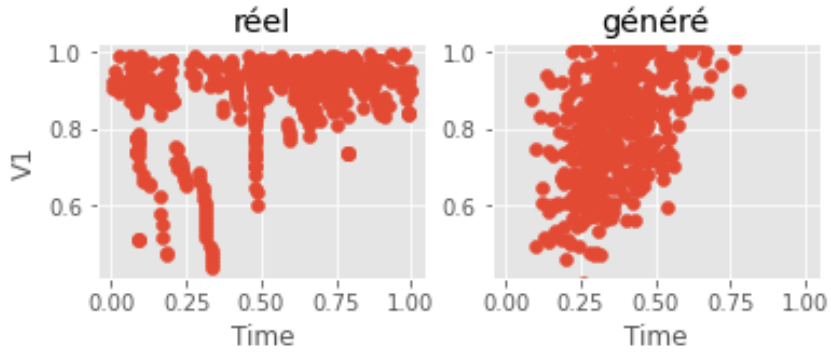
Les réseaux génératifs adverses (GAN) sont également une alternative permettant de régler le déséquilibre dans notre ensemble de données. Dans cette logique, nous avons entraîné deux architectures de réseaux génératifs adverses qui sont les suivants :

- ❖ GAN : Le GAN original ("vanille") [35]
- ❖ CGAN : Une version conditionnelle du GAN original qui utilise les étiquettes de classe [54]

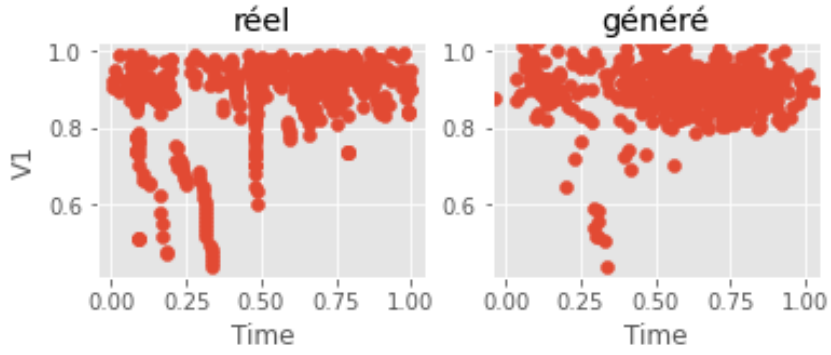
Nos deux architectures de GAN sont entraînées uniquement sur les transactions frauduleuses de chacun de nos deux jeux de données, car notre objectif est de pouvoir générer uniquement des données synthétiques frauduleuses. Nos deux architectures ont toutes trois couches cachées et prennent en paramètres un bruit aléatoire d'une dimension de 30 pour notre premier jeu de données et d'une dimension 22 pour notre deuxième jeu de données. À partir du bruit aléatoire, nos deux architectures génèrent

des données synthétiques de fraudes. Les figures 7.15 et 7.16 présentent la différence entre les données générées et réelles à partir de l'itération 1 à itération 5000 pour nos deux architectures. Une analyse poussée des figures 7.15 et 7.16 nous montre que les données générées sont de plus en plus précises au fur et à mesure des itérations, également, il ressort que c'est l'architecture CGAN qui génère de meilleures données synthétiques après les 5000 itérations.

Pour nous assurer de la qualité des données générées par notre CGAN, nous avons utilisé la bibliothèque **"TableEvaluator"** qui permet d'évaluer la similitude d'un ensemble de données synthétisées avec des données réelles. TableEvaluator via sa méthode de calcul de corrélation entre les données de fraude réelles et ceux qui ont été générés nous a donné comme résultats 99.45% sur le premier jeu de données et 90.23% sur notre deuxième jeu de données, pour faire ce calcul la bibliothèque utilise la méthode Spearman's pour sa résistance aux valeurs aberrantes et aux écarts entre les plages des valeurs.

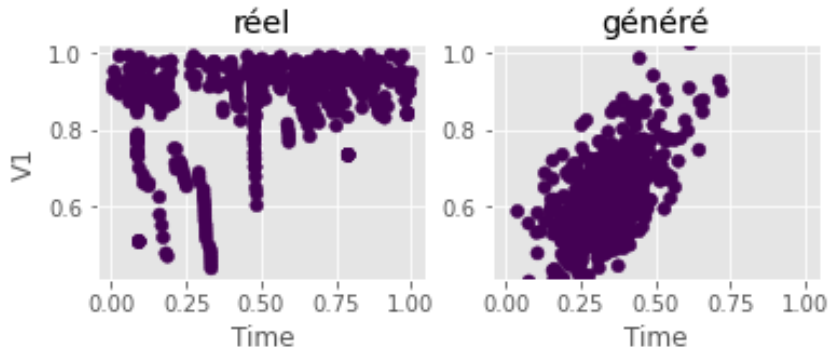


(a) Vanila gan première itération

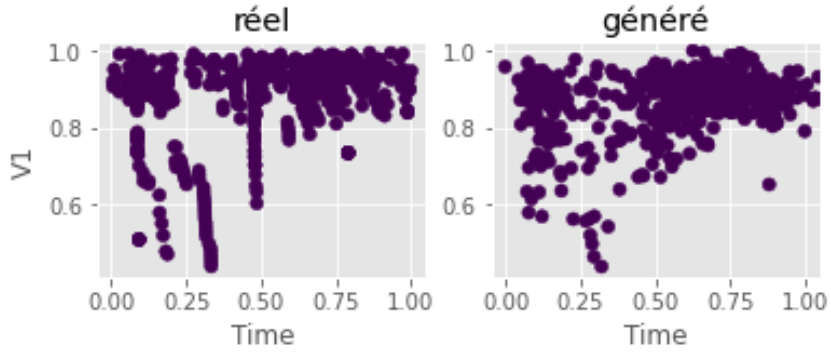


(b) Vanila gan après 5000 itérations

FIGURE 5.19 – Les étapes d'itération du Vanila gan



(a) Cgan première itération



(b) Cgan après 5000 itérations

FIGURE 5.20 – Les étapes d'itération du CGAN

5.4.6 Sélection de l'algorithme de génération des données

La technique de suréchantillonnage synthétique des minorités (SMOTE) bien que générant de nouvelles données basées sur les voisins les plus proches, elle a cependant comme inconvénient majeur de ne pas prendre en considération uniquement l'ensemble des données de la classe minoritaire, cela peut conduire à la génération de données artificielles biaisées.

Les réseaux génératifs adverses (GAN) permettent de pallier aux limitations de l'algorithme SMOTE en prenant en compte la distribution de l'ensemble de données d'entraînement lors de la génération de nouvelles données. Également il est possible d'affiner les performances des réseaux génératifs adverses contrairement à SMOTE, cette flexibilité s'explique par la possibilité de pouvoir contrôler un champ spécifique en réglant de nombreux paramètres. L'un des avantages des réseaux génératifs adverses demeure la capacité à pouvoir générer des données de haute qualité même avec de très petits échantillons. Nos conclusions sont confirmées par les figures 5.21 et 5.22 qui indiquent clairement la différence de qualité entre les données de fraude générées par l'algorithme SMOTE et celles produites par les réseaux générateurs adverses.

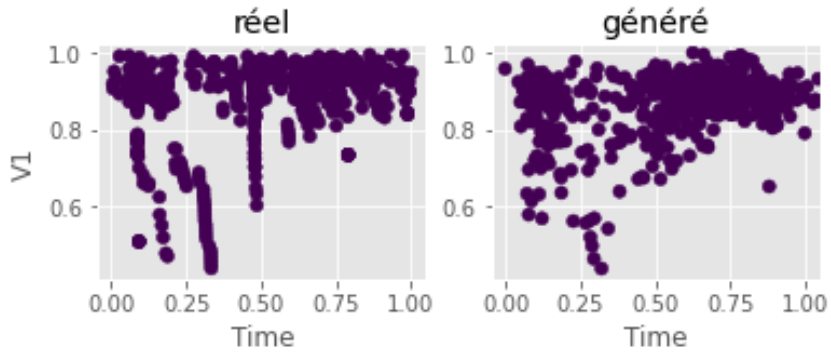


FIGURE 5.21 – Cgan après 5000 itérations

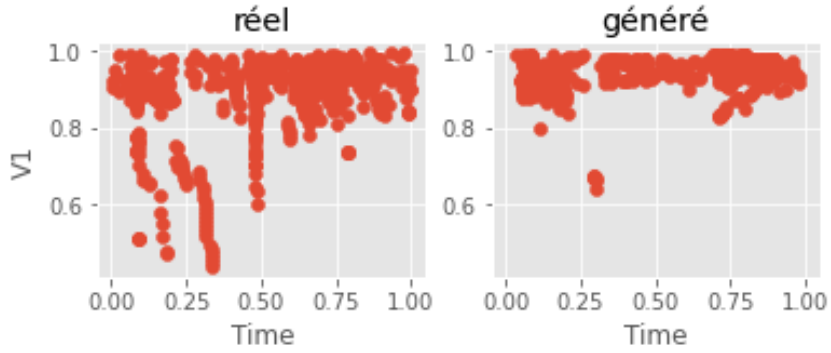


FIGURE 5.22 – Données de fraude générées par SMOTE

5.4.7 Entrainement des Autoencodeurs double

Afin d'accroître la diversité de nos ensembles de données, nous allons entraîner deux Autoencodeurs, ce qui nous permettra de reconnaître les caractéristiques essentielles des données de fraude et des données normales [27],[79]. Notre premier Autoencodeur est entrainé à partir des données représentant les transactions normales. En observant la figure 5.23 présentant les résultats d'entrainement du premier Autoencodeur, nous pouvons voir que la perte(erreur) du modèle diminue aux fur et à mesure des époques à la fois sur les données d'entrainement que sur les données de validation. En analysant la figure 5.24 nous remarquons que les données reconstruites par notre modèle sont similaires aux données de test avec une erreur minime, cela est également confirmé par la figure 5.25 ou on peut voir que l'erreur est minime. Il est important de rappeler que nous avons divisé les transactions normales en trois parties qui sont les suivantes :

- ❖ donnée d'entrainement
- ❖ donnée de validation
- ❖ donnée de teste

L'Autoencodeur de fraude va être entrainé en deux étapes , la première étape consiste a entrainer l'Autoencodeur avec uniquement les données de fraude initiale les

résultats obtenus seront par la suite comparés aux seconds entrainements de l'Autoencodeur dans lequel l'Autoencodeur est entrainé avec des données augmentées qui sont le résultat de la concaténation entre les données de fraudes générer par notre réseau génératif adverse (`g_f`) avec les données de fraude initiales de notre ensemble de données(`r_f`). En analysant la figure 5.26 présentant les résultats d'entrainement de la première variation de l'Autoencodeur nous remarquons que la perte(erreur) du modèle diminue faiblement et cela malgré le grand nombre d'époques pendant lequel le modèle est entrainé cela nous montre que le modèle a du mal apprendre les données. Cette faible performance du modèle s'explique par la faible quantité de données d'entrainement et de validation. Également la figure 5.27 nous montre que l'erreur de reconstruction des données est très élevée montrant que les données reconstruites ne sont pas similaires aux données de fraudes initiales , cela nous est confirmé par la figure 5.28 dans la quel nous voyons que la perte de teste est très élevés.

En analysant la figure 5.29 présentant les résultats de l'Autoencodeur entrainer sur les données de fraudes augmenter, nous remarquons que l'erreur(perte) du modèle diminue aux fure et à mesure des époques ce qui montre que le modèle est à mesure d'apprendre les caractéristiques essentielles propres aux données de fraude, contrairement a la figure 5.26 dans ou on constate que l'erreur (perte) du modèle diminue faiblement. L'analyse de la figure 5.30 montre que l'erreur de reconstruction est faible, ce qui est en contraste avec la figure 5.27 dans laquelle nous constatons les données reconstruites diverge fortement des données de test. Les résultats de notre analyse nous permettent d'affirmer que la performance du modèle s'est fortement améliorée grâce à l'entraînement avec les données de fraude augmentées, cela est également confirmer par la faible perte de test que nous montre la figure 5.31. L'Autoencodeur normal et celui de fraude seront utilisés pour générer de nouvelles données supplémentaires. Ces nouvelles données seront par la suite concaténées avec notre ensemble de données pour entraîner notre modèle de classification. L'ensemble de données finales sera ainsi très diversifié et permettra d'améliorer la performance ainsi que la robustesse de notre modèle de classification.

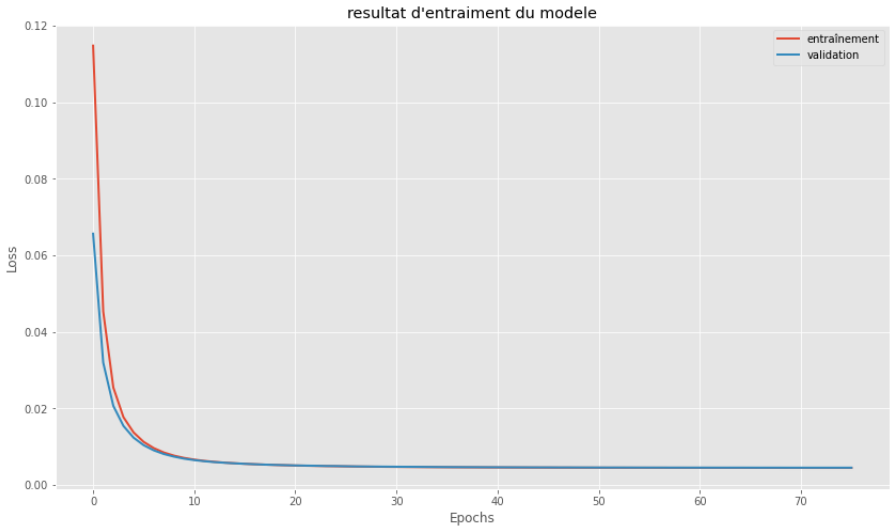


FIGURE 5.23 – Résultats d'entrainement de l'Autoencodeur à partir des échantillons de transaction normale

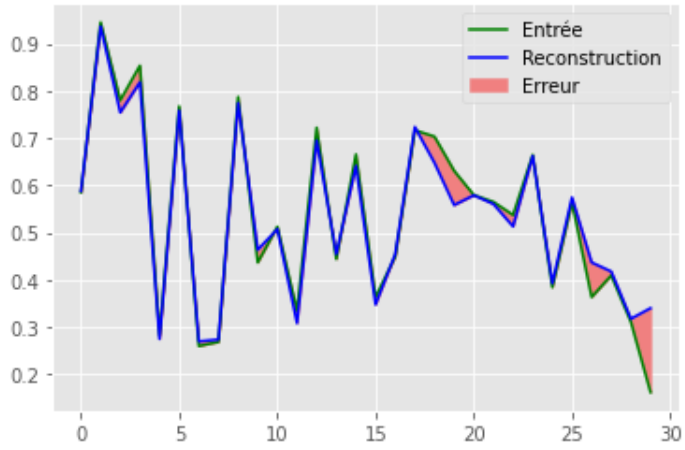


FIGURE 5.24 – Erreur de reconstruction Autoencodeur normale

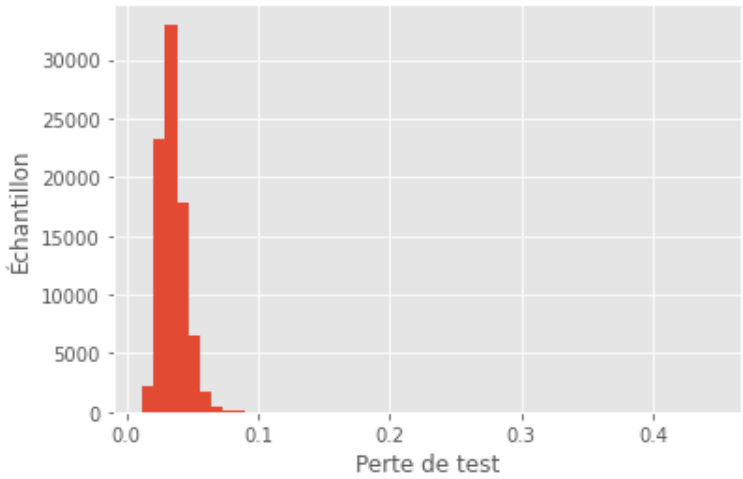


FIGURE 5.25 – Perte de test de l'Autoencodeur normale

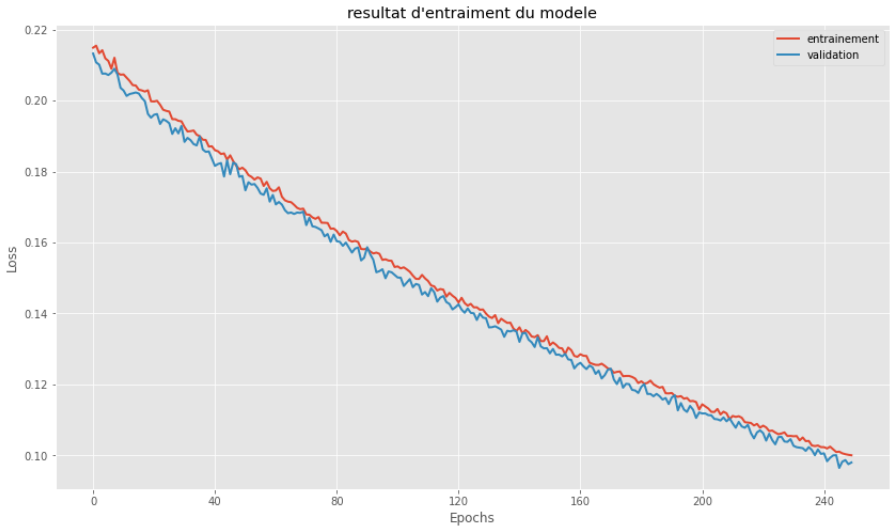


FIGURE 5.26 – Résultats d'entrainement de l'Autoencodeur à partir des échantillons de transaction de fraude initiales de notre ensemble de données

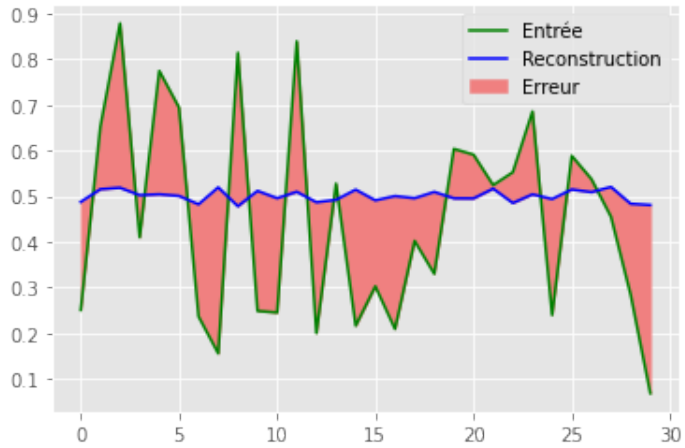


FIGURE 5.27 – Erreur de reconstruction Autoencodeur entrainer à partir des échantillons de transaction de fraude uniquement

FIGURE 5.28 – Perte de test de l'Autoencodeur entrainer à partir des échantillons de transaction de fraude uniquement

FIGURE 5.29 – Résultats d'entrainement de l'Autoencodeur à partir des Les données de fraude augmentées

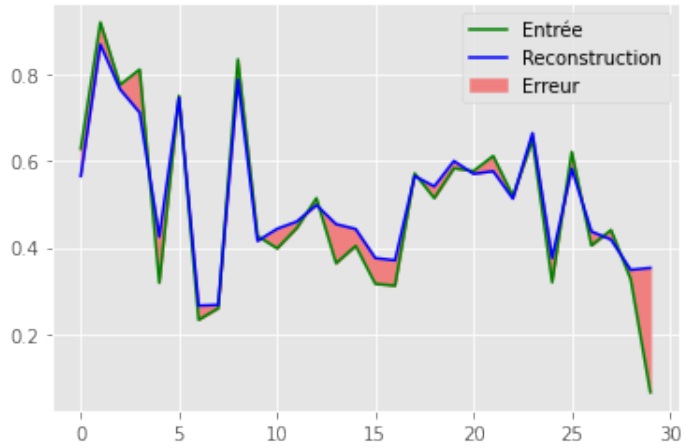


FIGURE 5.30 – Erreur de reconstruction Autoencodeur de fraude augmentées

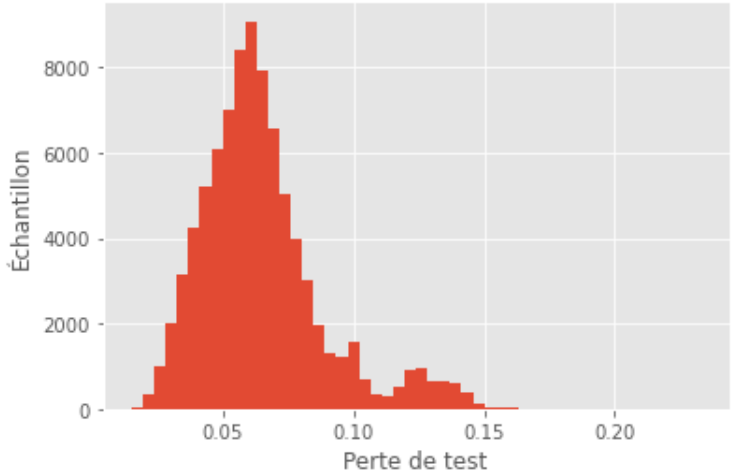


FIGURE 5.31 – Perte de test de l'Autoencodeur de fraude augmentées

5.4.8 Entraînement et test de la performance des différents modèles sur nos ensembles de données augmentées

Dans cette section, nous allons évaluer de nouveau les performances de nos cinq algorithmes, mais cette fois-ci, ce sera sur nos deux jeux de données qui ont été augmentés(équilibrés) contrairement aux tests précédents qui avaient été effectués sur nos jeux de données initiales (déséquilibrés). Ayant donc effectué les différents tests de performance, nous sommes arrivés aux résultats contenus dans les tableaux 5.3 et 5.4 qui présentent les performances de nos cinq algorithmes sur chacun de nos jeux de données augmentées(équilibrés).

En examinant les tableaux 5.3 et 5.4, nous constatons une nette amélioration des performances de nos différents modèles par rapport aux résultats obtenus suite aux entraînements et aux tests effectués sur nos jeux de données déséquilibrés. Le pourcentage élevé de la métrique de rappel nous montre que nos différents modèles sont capables d'identifier les transactions frauduleuses des transactions normales. Pour mieux illustrer cette analyse, intéressons-nous à l'algorithme XGBoost, nous constatons une amélioration de l'ordre de 22.96% et 28.73% de la métrique rappel sur nos deux jeux de données, cela est également confirmé par les figures 5.33 et 5.35 sur lesquelles nous constatons que l'erreur du modèle XGBoost diminue à la fois sur les données d'entraînement et de test ce qui montre que notre modèle s'adapte aux données d'apprentissage. L'amélioration de la performance de notre modèle est confirmée par le pourcentage élevé du scoreF1 ou nous notons pour tous nos algorithmes une amélioration moyenne de plus de 15% sur tous nos différents modèles, ce constat est particulièrement vrai pour notre deuxième jeu de données sur lequel la Régression logistique et SVM ont connu amélioration drastique de leurs scoresF1, rappel et précisions.

Les résultats de nos différents tests sur nos deux jeux de données nous montrent que globalement ce sont les algorithmes XGBoost et RandomForest qui sont les plus performants, tandis que la Régression logistique est l'algorithme qui s'avère le moins performant sur nos deux jeux de données. Nous constatons que globalement nos modèles performent légèrement mieux sur notre premier jeu de données. Les conclusions des résultats des analyses des tableaux 5.3 et 5.4 nous confirment une nette amélioration de la performance de nos cinq modèles(figure 5.32 et 5.34) quand ils sont entraînés sur des jeux de données équilibrées , cela confirme l'impact du déséquilibre des données sur la performance des modèles d'apprentissage automatique [1],[43],[56]. La prochaine étape va consister pour nous à tester la robustesse de nos différents modèles, pour cela nous allons implémenter une attaque adverse sur chacun de nos modèles.

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	99.98%	99.97%	99.97%	99.97%	99.97%
RandomForest	99.98%	99.98%	99.97%	99.98%	99.97%
LightGBM	99.94%	99.94%	99.93%	99.94%	99.93%
SVM	99.97%	99.97%	99.97%	99.97%	99.97%
Régression logistique	99.88%	99.93%	99.90%	99.91%	99.90%

TABLE 5.3 – Performance des différents algorithmes sur le premier jeu de données augmentées

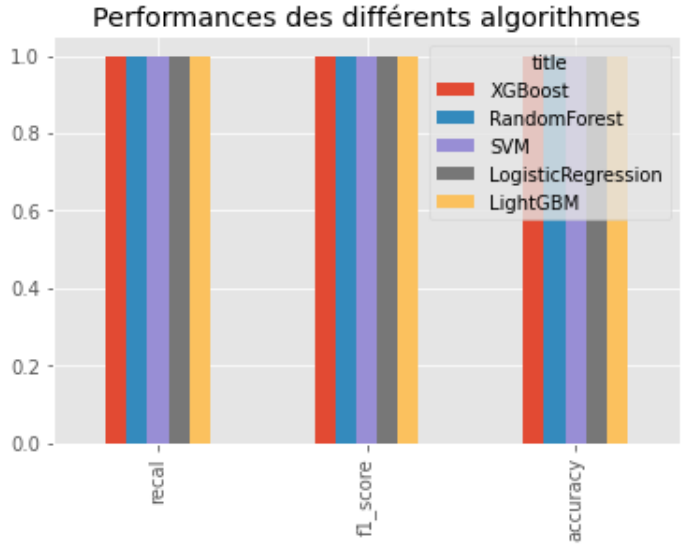


FIGURE 5.32 – Le graphe de performance des différents algorithmes sur le premier jeu de données augmentées

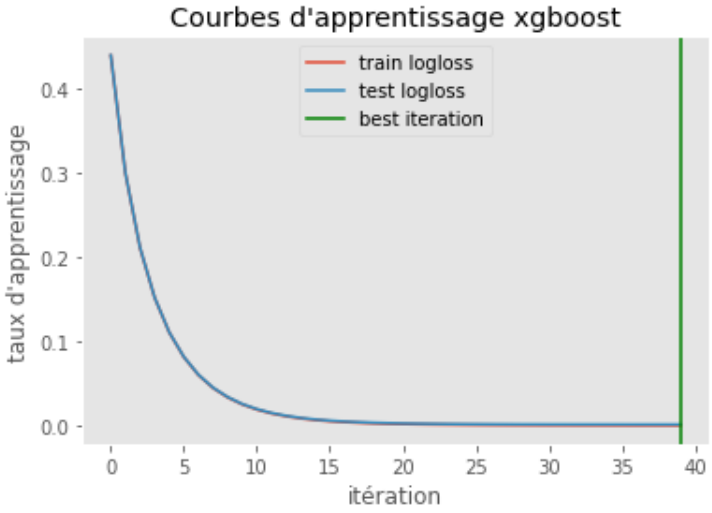


FIGURE 5.33 – Courbes d'apprentissage xgboost sur le premier jeu de données augmentées

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	99.71%	99.88%	99.82%	99.77%	99.85%
RandomForest	99.67%	99.88%	99.80%	99.77%	99.84%
LightGBM	99.66%	99.82%	99.78%	99.74%	99.81%
SVM	98.95%	99.87%	99.44%	99.41%	99.58%
Régression logistique	98.16%	99.64%	98.98%	98.90%	99.22%

TABLE 5.4 – Performance des différents algorithmes sur le deuxième jeu de données augmentées

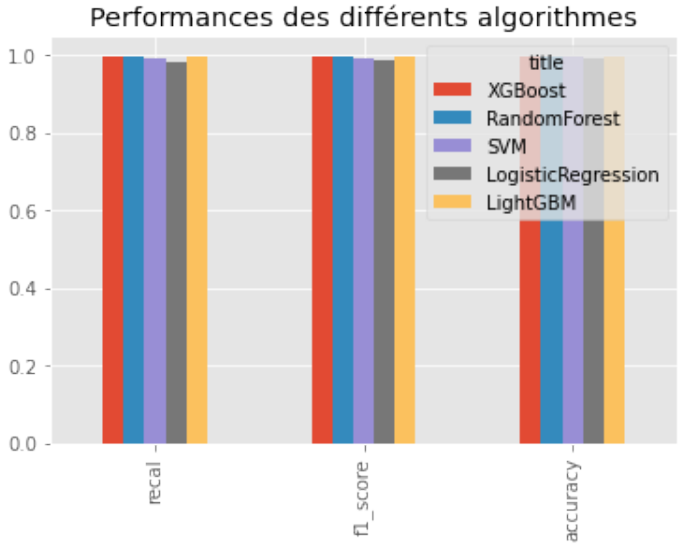


FIGURE 5.34 – Le graphe de performance des différents algorithmes sur le deuxième jeu de données augmentées

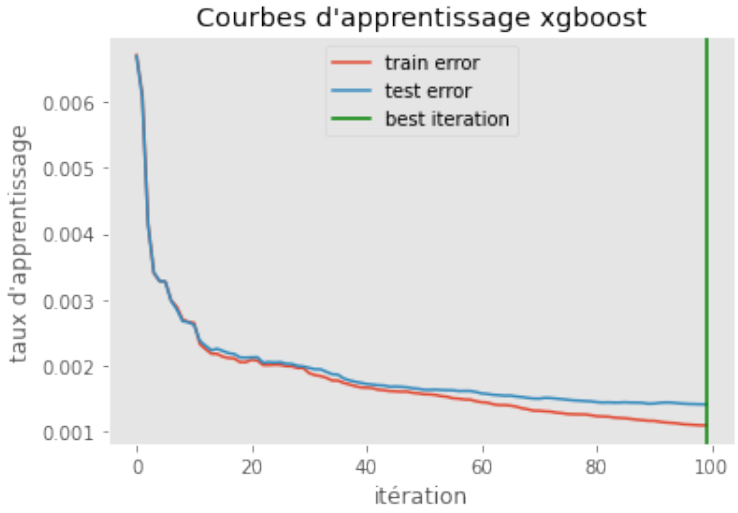


FIGURE 5.35 – Courbes d'apprentissage xgboost sur le deuxième jeu de données augmentées

5.4.9 Tests et amélioration de la robustesse des modèles

Dans cette section, nous allons tester la robustesse de nos cinq modèles, pour cela nous avons utilisé la bibliothèque "Adversarial Robustness Toolbox" pour implémenter une attaque adverse. L'attaque adverse que nous avons choisie est connue sous le nom de ZOO (Zeroth Order Optimization), notre choix a été motivé en raison de la performance de l'algorithme ZOO, également il est intégré dans la bibliothèque "Adversarial Robustness Toolbox". Il s'agit d'une attaque de type boîte noire qui fonctionne sur le principe de la génération d'exemples contradictoires qui seront envoyés comme entrés à chacun de nos différents modèles afin de les induire en erreur dans leur classification [49]. Le principe de fonctionnement de l'attaque ZOO est amplement détaillé dans la section "Contributions" de notre document. Les tableaux 5.5 et 5.6 présentent les résultats qui ont été obtenus suite aux attaques adverses et feront l'objet d'analyses par la suite.

En analysant les tableaux 5.5 et 5.6 présentant les résultats des attaques de nos échantillons contradictoires sur nos cinq modèles, nous constatons globalement une baisse de la performance de nos cinq modèles. En examinant de plus près la métrique rappel, nous constatons une baisse significative pour nos cinq modèles sur nos deux jeux de données, si l'on considère le modèle XGBoost nous notons une baisse de 17.95% sur notre premier jeu de données et 21.62% sur notre deuxième jeu de données de la métrique rappel. Cette baisse de la métrique "rappel" nous indique une augmentation des faux négatifs qui résulte des perturbations entraînées par nos attaques adverses. Le même constat ressort quand nous nous intéressons au scoreF1 pour lequel nous notons une baisse du pourcentage qui est de l'ordre de 10% en moyenne sur notre premier jeu de donnée et de 10% sur notre deuxième jeu de donnée. Cette baisse du score F1 traduit une diminution de la précision et du rappel de nos modèles, cette analyse nous est confirmée par la métrique "rappel" et "précision" de nos différents modèles. Nous notons également une baisse de la métrique "Accuaracy", ce qui nous indique une baisse de la prédiction des résultats corrects par nos modèles. XGBoost est l'algorithme qui s'avère le plus robuste parmi nos algorithmes, en deuxième position nous avons RandomForest, LightGBM arrive en troisième position. La Régression logistique et SVM sont ceux ayant été le plus impactés par nos échantillons contradictoires, en effet nous constatons une hausse significative des faux négatifs pour ces deux algorithmes.

Les conclusions des analyses des tableaux 5.5 et 5.6 confirment une baisse significative de la performance de nos modèles sur les échantillons contradictoires comme l'illustrent également les figures 5.36 et 5.37. Bien qu'il existe plusieurs solutions pour contrer les attaques adverses, notre solution c'est l'entraînement adverse. Nous avons choisi cette solution pour son efficacité et sa simplicité, l'entraînement contradictoire consiste à introduire les exemples contradictoires au cours du processus d'entraînement de nos modèles [21]. En analysant les tableaux 5.7 et 5.8, nous constatons que suite à

l'entraînement adverse les métriques de performance de nos modèles se sont considérablement améliorées. Cette hausse de la performance des modèles nous confirme l'efficacité de l'entraînement adverse.

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	82.03%	90.34%	86.63%	85.98%	86.63%
RandomForest	82%	99.81%	90.92%	90.03%	90.92%
LightGBM	83.43%	89.54%	86.84%	86.38%	86.64%
SVM	82.03%	88.50%	85.69%	85.15%	85.69%
Régression logistique	91.37%	88.16%	89.55%	89.73%	89.55%

TABLE 5.5 – Performance des différents algorithmes sur nos échantillons adverses générés à partir du premier jeu de données augmenté

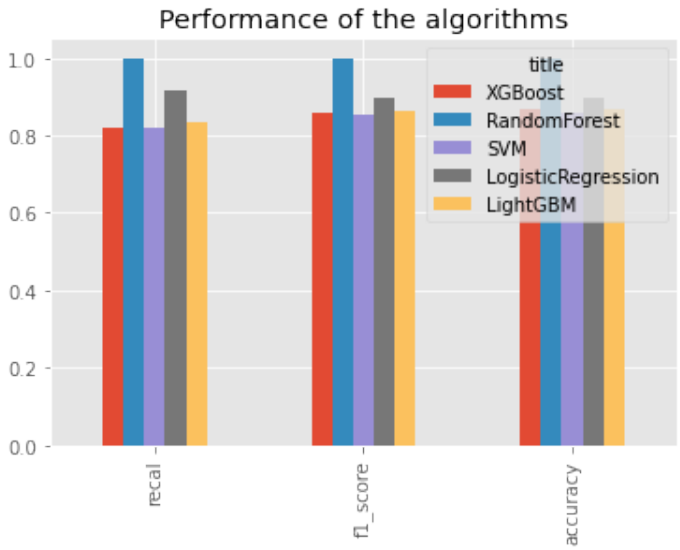


FIGURE 5.36 – Le graphe de performance des différents algorithmes sur nos échantillons adverses générés à partir du premier jeu de données augmenté

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	78.09%	92.87%	86.05%	84.84%	86.05%
RandomForest	75.82%	94.48%	85.69%	84.13%	85.69%
LightGBM	80.83%	93.31%	87.51%	86.62%	87.51%
SVM	81.29%	97.71%	89.69%	88.75%	89.69%
Régression logistique	81.20%	96.50%	89.13%	88.19%	89.13%

TABLE 5.6 – Performance des différents algorithmes sur nos échantillons adverses générés à partir du deuxième jeu de données augmenté

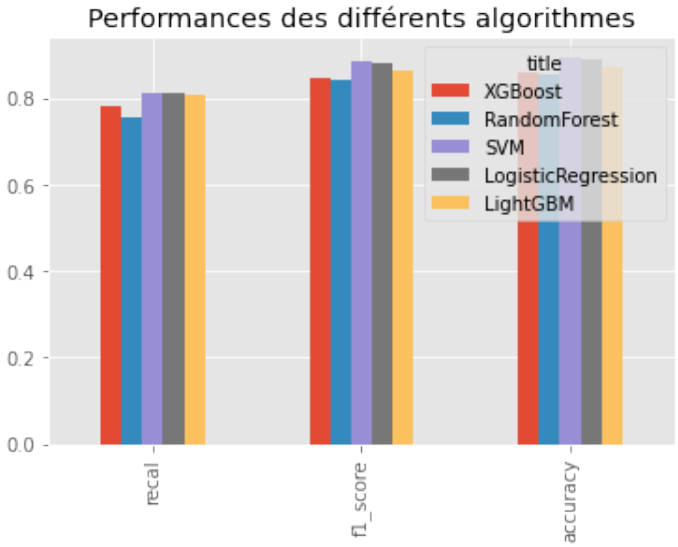


FIGURE 5.37 – Le graphe de performance des différents algorithmes sur nos échantillons adverses générés à partir du deuxième jeu de données augmenté

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	99.97%	99.96%	99.95%	99.96%	99.96%
RandomForest	99.98%	99.99%	99.98%	99.98%	99.98%
LightGBM	99.97%	99.92%	99.94%	99.94%	99.94%
SVM	99.73%	99.71%	99.72%	99.72%	99.72%
Régression logistique	98.72%	99.15%	98.93%	98.93%	98.93%

TABLE 5.7 – Performance des algorithmes réentraînée en incluant nos échantillons contradictoires dans notre premier ensemble de données augmenté

Algorithmes	**Rappel**	**Précision**	**RocAuc**	**ScoreF1**	**Accuracy**
XGBoost	99.87%	99.87%	99.87%	99.87%	99.87%
RandomForest	99.93%	99.96%	99.94%	99.94%	99.94%
LightGBM	99.53%	99.59%	99.56%	99.56%	99.56%
SVM	95.45%	95.56%	95.50%	95.50%	95.60%
Régression logistique	89.71%	91.75%	90.80%	90.72%	90.82%

TABLE 5.8 – Performance des algorithmes réentraînée en incluant nos échantillons contradictoires dans notre deuxième ensemble de données augmenté

Chapitre 6

Conclusion

Dans ce mémoire, nous avons étudié la problématique de la détection des fraudes sur les cartes de crédit dans des ensembles de données déséquilibrés tout en prenant en compte la robustesse du modèle proposé. Nous avons utilisé deux jeux de données publiques, dont le premier est composé de 31 caractéristiques et contient 284 807 transactions dont 492 sont des transactions frauduleuses qui représentent 0,172% de toutes les transactions. Le second jeu de données est composé de 22 caractéristiques et contient 1 296 675 transactions dont 7506 sont des transactions frauduleuses ce qui représente 0.0057% de toutes les transactions. La première étape de notre approche a consisté à résoudre le déséquilibre présent dans nos deux jeux de données, pour cela nous avons exploré des techniques de génération de données synthétiques comme SMOTE et GAN. Les algorithmes choisis pour effectuer nos tests sont XGBoost, RandomForest, LightGBM, SVM, et la Régression logistique. Nous avons utilisé les métriques d'évaluation courante, notamment l'Accuracy, la précision, le rappel, le scoreF1 et RocAuc. Les premières expérimentations ont été effectuées sur nos deux jeux de données déséquilibrés, il ressort que globalement sur nos deux jeux de données que ce sont les algorithmes XGBoost (Score F1 de 84.75% sur notre premier jeu de données et 77.73% sur notre deuxième jeu de données) et RandomForest (Score F1 de 84.41% sur notre premier jeu de données et 74.43% sur notre deuxième jeu de données) qui performent les mieux. Les tests effectués sur nos données équilibrées nous montrent une nette amélioration des métriques de performance, ce qui confirme l'impact qu'a le déséquilibre des données sur la performance des algorithmes d'apprentissage automatique. En considérant l'algorithme XGBoost on note une amélioration de 22.96% de la métrique rappel et une amélioration de 15.22% de la métrique ScoreF1 sur notre premier jeu de données quand il est entraîné sur des données équilibrées. Il ressort également de nos expérimentations que l'attaque ZOO (zeroth order optimisation) a impacté nos modèles, ce qui s'est matérialisé par une baisse de la performance des modèles attaqués. Nos expérimentations ont conclu que l'entraînement contradictoire s'avère une technique efficace pour contrer les attaques adverses, plus particulièrement l'attaque ZOO. Pour des travaux futurs permettant l'amélioration de la robustesse et de

la performance de notre solution, nous pouvons explorer la sélection des caractéristiques, l'utilisation de l'apprentissage fédéré pour l'entraînement des modèles, rendre le processus plus léger. Nous pourrons également tester d'autres types d'attaques adverses sur nos modèles, cela nous permettra d'effectuer les analyses et les discussions nécessaires pour contrer ces nouveaux types d'attaques.

Bibliographie

[1] Abdelaziz Abhishek. "Machine Learning for Imbalanced Data : Tackle Imbalanced Datasets Using Machine Learning and Deep Learning Techniques". In : (2023). doi : https://doi.org/10.15439/2023F3838..

[2] Kassim Pels Afriyie Jonathan Kwaku Tawiah. "A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions". In : 6 (2023). doi : 10.1016/j.dajour.2023.100163.

[3] Khairol Amali Bin Ahmad Ahmad. "Machine Learning and Big Data : Concepts, Algorithms, Tools and Applications". In : *The National Academies Press* (2020).

[4] Al-Sakib Khan Ahmed Mohiuddin Pathan. "Data analytics : concepts, techniques and applications". In : (2018), p. 210. url : https://uqtr.skillport.com/skillportfe/assetSummaryPage.action?assetid=RW$1391:_ss_book:138692#summary/BOOKS/RW$1391:_ss_book:138692.

[5] Fawaz Khaled Alarfaj et al. "Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms". In : *IEEE Access* 10 (2022), p. 39700-39715. doi : 10.1109/ACCESS.2022.3166891.

[6] Amazon. *What is Hadoop ?* https://aws.amazon.com/fr/emr/details/hadoop/what-is-hadoop/.

[7] Nawal Badawy Arafa Ahmed El-Fishawy. "RN-Autoencoder : Reduced Noise Autoencoder for classifying imbalanced cancer genomic data". In : *Journal of Biological Engineering* (2023). doi : 10.1186/s13036-022-00319-3.

[8] Martin Arjovsky, Soumith Chintala et Léon Bottou. *Wasserstein GAN*. 2017. doi : 10.48550/ARXIV.1701.07875. url : https://arxiv.org/abs/1701.07875.

[9] Rohan Singh Lee Arora Vinay Leekha. "Facilitating User Authorization from Imbalanced Data Logs of Credit Cards Using Artificial Intelligence". In : (2020), p. 8885269. doi : 10.1155/2020/8885269.

[10] Bijan Ashtiani Matin Raahemi. "Intelligent Fraud Detection in Financial Statements Using Machine Learning and Data Mining : A Systematic Literature Review". In : *IEEE Access* (2022). doi : 10.1109/ACCESS.2021.3096799.

[11] *Asian Dataset,Credit Card Payments Default Online.Available : https ://www.kaggle.com/code/avikpaul4u/credit- card-payments-default/notebook.*

[12] Waldo Naiouf Basgall María José Hasperué. "SMOTE-BD : An Exact and Scalable Oversampling Method for Imbalanced Classification in Big Data". In : (2018). URL : https://web-s-ebscohost-com.biblioproxy.uqtr.ca/ehost/pdfviewer/pdfviewer?vid=0&sid=9a6b604f-2425-4da0-9d7f-fe7e73d068f2%40redis.

[13] Wawrowski Biczyk. "Towards Automated Detection of Adversarial Attacks on Tabular Data". In : (2023). DOI : https://doi.org/10.15439/2023F3838..

[14] Bootcamp. *What Is Data Mining : Definition, Examples, Tools, and Techniques (For Beginners).* https://bootcamp.pe.gatech.edu/blog/what-is-data-mining/.

[15] Bernardo Branco et al. "Interleaved Sequence RNNs for Fraud Detection". In : *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining.* KDD '20. ACM, août 2020. DOI : 10.1145/3394486.3403361. URL : http://dx.doi.org/10.1145/3394486.3403361.

[16] Tom B. Brown et al. *Adversarial Patch.* 2018. arXiv : 1712.09665 [cs.CV]. URL : https://arxiv.org/abs/1712.09665.

[17] Centre antifraude du Canada. *Répercussions de la fraude depuis le début de l'année.* https ://antifraudcentre-centreantifraude.ca/index-fra.htm.

[18] Gendarmerie royale du Canada. *Mois de la prévention de la fraude 2023 : Les pertes liées à la fraude atteignent un nouveau record historique au Canada.* https ://www.rcmp-grc.gc.ca/fr/nouvelles/2023/mois-prevention-fraude-2023-pertes-liees-a-fraude-atteignent-nouveau-record.

[19] M Wang Carlsson C Heikkila. "Fuzzy c-means for fraud detection in large transaction data sets". In : *IEEE International Conference on Fuzzy Systems* (2018). DOI : 10.1109/FUZZ-IEEE.2018.8491498.

[20] Francesco Cartella et al. *Adversarial Attacks for Tabular Data : Application to Fraud Detection and Imbalanced Data.* 2021. arXiv : 2101.08030 [cs.CR].

[21] Dionna National Academies of Sciences Casola Linda Clare Ali. "Robust machine learning algorithms and systems for detection and mitigation of adversarial attacks and anomalies : proceedings of a workshop". In : *The National Academies Press* (2019).

[22] N. V. Chawla et al. "SMOTE : Synthetic Minority Over-sampling Technique". In : *Journal of Artificial Intelligence Research* 16 (juin 2002), p. 321-357. ISSN : 1076-9757. DOI : 10.1613/jair.953. URL : http://dx.doi.org/10.1613/jair.953.

[23] Tianqi CHEN et Carlos GUESTRIN. "XGBoost : A Scalable Tree Boosting System". In : *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. ACM, août 2016. DOI : 10.1145/2939672.2939785. URL : http://dx.doi.org/10.1145/2939672.2939785.

[24] Kim CY CHOI JE Seol DH. "Generative Adversarial Network-Based Fault Detection in Semiconductor Equipment with Class-Imbalanced Data. Sensors (Basel)". In : (2023). DOI : 10.3390/s23041889. URL : https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9967967/.

[25] CLEARLYPAYMENTS. *Credit Card Fraud in 2023*. https://www.clearlypayments.com/blog/credit-card-fraud-in-2023.

[26] DATA.WORLD. *Fraud datasets available on data.world.* https://data.world/datasets/fraud.

[27] Raja Giryes DOR BANK Noam Koenigstein. "Autoencoders". In : (2021).

[28] Li Guo DU Haichao Lv. "AutoEncoder and LightGBM for Credit Card Fraud Detection Problems". In : 15 (2023), p. 870. DOI : 10.3390/sym15040870.

[29] Khaleel Khairol Amali Bin DULHARE Uma N Ahmad. "Machine learning and big data : concepts, algorithms, tools and applications". In : (2020).

[30] Ebenezer Mienye ESENOGHO. "A Neural Network Ensemble With Feature Engineering for Improved Credit Card Fraud Detection". In : *IEEE Access* (2022). DOI : 10.1109/ACCESS.2022.3148298.

[31] Fernando FONSECA Joao Bacao. "Geometric SMOTE for imbalanced datasets with nominal and continuous features". In : (2023). URL : https://www-sciencedirect-com.biblioproxy.uqtr.ca/search?authors=fonseca&date=2023&docId=0957-4174&qs=Geometric%20SMOTE%20for%20imbalanced%20datasets%20with%20nominal%20and%20continuous%20features.

[32] GEEKSFORGEEKS. *Challenges of Data Mining*. https://www.geeksforgeeks.org/challenges-of-data-mining/.

[33] GEEKSFORGEEKS. *K-Nearest Neighbor(KNN) Algorithm*. URL : https://www.geeksforgeeks.org/k-nearest-neighbours/.

[34] Ian J. GOODFELLOW, Jonathon SHLENS et Christian SZEGEDY. "Explaining and Harnessing Adversarial Examples". In : (2015). arXiv : 1412.6572.

[35] Ian J. GOODFELLOW et al. *Generative Adversarial Networks*. 2014. DOI : 10.48550/ARXIV.1406.2661. URL : https://arxiv.org/abs/1406.2661.

[36] F GORUNESCU. "Data Mining : Concepts, Models and Techniques". In : (2011). DOI : https://doi.org/10.1007/978-3-642-19721-5..

[37] Ashraf Hassanien Aboul Ella Darwish. "Machine learning and big data analytics paradigms : analysis, applications and challenges". In : (2021). URL : https://uqtr.skillport.com/skillportfe/assetSummaryPage.action?assetid=RW$1391:_ss_book:138692#summary/BOOKS/RW$1391:_ss_book:138692.

[38] Ashraf Hassanien Aboul Ella Darwish. "Machine learning and big data analytics paradigms : analysis, applications and challenges". In : *Annalen der Physik* (2021). DOI : 10.1007/978-3-030-59338-4.

[39] Yawney Hilal Gadsden. "Financial Fraud : A Review of Anomaly Detection Techniques and Recent Advances." In : (2022). DOI : https://doi.org/10.1016/j.eswa.2021.116429..

[40] IBM. https://www.kaggle.com/datasets/ealtman2019/ibm-transactions-for-anti-money-laundering-aml.

[41] IBM. *What is the k-nearest neighbors (KNN) algorithm ?* URL : https://www.ibm.com/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20(KNN,used%20in%20machine%20learning%20today..

[42] Javatpoint. *K-Nearest Neighbor(KNN) Algorithm for Machine Learning*. https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning.

[43] Khoshgoftaar John Taghi. "Evaluating Classifier Performance with Highly Imbalanced Big Data. Journal of Big Data". In : (2023). DOI : https://doi.org/10.1186/s40537-023-00724-5..

[44] S Joshi. "Abstract data set for credit card fraud detection". In : (2020).

[45] Kaggle. *Credit Card Fraud Detection*. https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud. 2018.

[46] Mohammed M. Khaled et Zaher AL Aghbari. "ccfDetector : Utilizing GAN and Deep Learning for Credit Card Fraud Detection". In : *2023 Advances in Science and Engineering Technology International Conferences (ASET)*. 2023, p. 1-6. DOI : 10.1109/ASET56582.2023.10180825.

[47] UM Learning. *Default of credit card clients dataset*. 2016.

[48] Tzu-Hsuan Lin et Jehn-Ruey Jiang. "Credit Card Fraud Detection with Autoencoder and Probabilistic Random Forest". In : *Mathematics* 9.21 (2021). ISSN : 2227-7390. DOI : 10.3390/math9212683. URL : https://www.mdpi.com/2227-7390/9/21/2683.

[49] Kailkhura Liu Chen. "A Primer on Zeroth-Order Optimization in Signal Processing and Machine Learning : Principals, Recent Advances, and Applications". In : (2020). DOI : https://doi.org/10.1109/MSP.2020.3003837..

[50] MACHINELEARNINGMASTERY. *How to Develop a Wasserstein Generative Adversarial Network*. https://machinelearningmastery.com/how-to-code-a-wasserstein-generative-adversarial-network-wgan-from-scratch.

[51] Aleksander MADRY et al. "Towards Deep Learning Models Resistant to Adversarial Attacks". In : (2019). arXiv : 1706.06083.

[52] Santosh Kumar MAJHI. "Fuzzy clustering algorithm based on modified whale optimization algorithm for automobile insurance fraud detection". In : (2019). DOI : 10.1007/s12065-019-00260-3.

[53] Soumith Chintala et Léon Bottou MARTIN ARJOVSKY. "Wasserstein GAN". In : (2017). DOI : 10.48550/ARXIV.1701.07875.

[54] Mehdi MIRZA et Simon OSINDERO. *Conditional Generative Adversarial Nets*. 2014. DOI : 10.48550/ARXIV.1411.1784. URL : https://arxiv.org/abs/1411.1784.

[55] Sidharth MISHRA et al. "Principal Component Analysis". In : *International Journal of Livestock Research* (jan. 2017), p. 1. DOI : 10.5455/ijlr.20170415115235.

[56] Sangeeta MITTAL et Shivani TYAGI. "Performance Evaluation of Machine Learning Algorithms for Credit Card Fraud Detection". In : *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*. 2019, p. 320-324. DOI : 10.1109/CONFLUENCE.2019.8776925.

[57] OPENGENUS. *Types of Generative Adversarial Networks (GANs)*. https://iq.opengenus.org/types-of-gans/.

[58] Kumar PETER Manoj. "Blockchain and Machine Learning Approaches for Credit Card Fraud Detection". In : (2023). DOI : https://doi.org/10.1109/ICSSIT55814.2023.10060999..

[59] Smruthi PORWAL Utkarsh Mukund. "Credit Card Fraud Detection in E-Commerce". In : *IEEE Access* (2019). DOI : 10.1109/TrustCom/BigDataSE.2019.00045.

[60] Kazi Tamzid Akhter Md Hasib PRANTO Tahmid Hasan. "Blockchain and Machine Learning for Fraud Detection : A Privacy-Preserving and Adaptive Incentive Based Approach". In : (2022). DOI : https://doi.org/10.1109/ACCESS.2022.3198956..

[61] Rohit RAJA. "Data mining and machine learning applications". In : *Wiley Scrivener Publishing* (2022). DOI : 10.1002/9781119792529.

[62] Vasile Palade RUKSHAN BATUWITA. "CLASS IMBALANCE LEARNING METHODS FOR SUPPORT VECTOR MACHINES". In : (2023).

[63] Iqbal H. SARKER. "Machine Learning : Algorithms, Real-World Applications and Research Directions". In : (2021). ISSN : 2662-995X. DOI : 10.1007/s42979-021-00592-x.

[64] Soraya SEDKAOUI. "Data analytics and big data". In : (2018). DOI : 10.1002/9781119528043. URL : https://public.ebookcentral.proquest.com/choice/publicfullrecord.aspx?p=5401178.

[65] Singh SEHRAWAT Deepthi. "Auto-Encoder and LSTM-Based Credit Card Fraud Detection". In : 4 (2023). DOI : 10.1007/s42979-023-01977-w.

[66] M SEKAR. "Machine Learning for Auditors : Automating Fraud Investigations through Artificial Intelligence". In : *Apress : New York, NY* (2022).

[67] Ganesh SHARMA Abhilash Raj. "Credit Card Fraud Detection Using Deep Learning Based on Auto-Encoder". In : 2022. DOI : 10.1051/itmconf/20225001001.

[68] *Simulated Credit Card Transactions generated using Sparkov*. 2020.

[69] G. M. Suhas JAIN et al. "A Novel Approach in Credit Card Fraud Detection System Using Machine Learning Techniques". In : *2021 International Conference on Forensics, Analytics, Big Data, Security (FABS)*. T. 1. 2021, p. 1-5. DOI : 10.1109/FABS52071.2021.9702672.

[70] Cheng Kuihua TINGFEI Huang Guangquan. "Using Variational Auto Encoding in Credit Card Fraud Detection". In : 2020. DOI : 10.1109/ACCESS.2020.3015600.

[71] TOWARDSDATASCIENCE. *Support Vector Machines(SVM) — An Overview*. https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989.

[72] TOWARDSDATASCIENCE. *Understanding Data Preprocessing*. https://towardsdatascience.com/data-preprocessing-e2b0bed4c7fb.

[73] TRANSUNION. *Digital Transactions in Canada Suspected Fraudulent in H1 2023*. https://newsroom.transunion.ca/nearly-half-49-of-canadians-said-they-were-recently-targeted-by-fraud-around-1-in-20-digital-transactions-in-canada-suspected-fraudulent-in-h1-2023-reveals-transunion-canada-analysis/.

[74] Rishabh TYAGI, Ravi RANJAN et S. PRIYA. "Credit Card Fraud Detection Using Machine Learning Algorithms". In : *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. 2021, p. 334-341. DOI : 10.1109/I-SMAC52330.2021.9640822.

[75] *UCI Machine Learning Repository. (2017, Nov. 29). Statlog (Australian credit approval) dataset Online*. 2017.

[76] WIKIPEDIA. *Apache Cassandra*. https://en.wikipedia.org/wiki/Apache_Cassandra.

[77] WIKIPEDIA. *Apache Kafka*. https://en.wikipedia.org/wiki/Apache_Kafka.

[78] WIKIPEDIA. *Athletics at the 2008 Summer Olympics – Men's 200 metres*.

[79] WIKIPEDIA. *Autoencoder*. https://en.wikipedia.org/wiki/Autoencoder.

[80] WIKIPEDIA. *Generative adversarial network*. https://en.wikipedia.org/wiki/Generative_adversarial_network.

[81] WIKIPEDIA. *Minimax*. https://en.wikipedia.org/wiki/Minimax.

[82] WIKIPEDIA. *Multiple instance learning*. https://en.wikipedia.org/wiki/Multiple_instance_learning.

[83] WIKIPEDIA. *Naive Bayes classifier*. https://en.wikipedia.org/wiki/Naive_Bayes_classifier.

[84] WIKIPEDIA. *Oversampling and undersampling in data analysis*. https://en.wikipedia.org/wiki/Oversampling_and_undersampling_in_data_analysis.

[85] WIKIPEDIA. *Wasserstein GAN*. https://en.wikipedia.org/wiki/Wasserstein_GAN.

[86] WIKIPEDIA. *Wikipedia :Adversarial machine learning*. https://en.wikipedia.org/wiki/Adversarial_machine_learning#Mechanisms.

[87] WIKIPEDIA. *Wikipedia :Data analysis*. https://en.wikipedia.org/wiki/Data_analysis.

[88] Hongyan Welsch WU Ensen Cui. "Dual Autoencoders Generative Adversarial Network for Imbalanced Classification Problem". In : (2020). DOI : 10.1109/ACCESS.2020.2994327.

[89] Arun George ZACHARIAH. "Adversarial Attacks with Carlini Wagner Approach". In : 2023.