



Centre d'Etudes Doctorales : Sciences et Techniques de l'Ingénieur

N° d'ordre : 46/2021

THESE DE DOCTORAT

Présentée par

Mme : Salma EL HAJJAMI

Discipline : Informatique

Spécialité : Informatique

Sujet de la thèse : Élaboration d'une technique d'apprentissage automatique pour la détection de fraudes par carte de crédit : Application sur les données massives déséquilibrées.

Formation Doctorale : Sciences de l'ingénieur Sciences Physiques, Mathématiques et Informatique.

Thèse présentée et soutenue le samedi 26 juin 2021 à 11h au Centre de conférences devant le jury composé de :

Nom Prénom	Titre	Établissement	
Nour El Houda CHAOUI	PES	École Nationale des Sciences Appliquées de Fès	Président
Ahmed ZELLOU	PES	École Nationale Supérieure d'Informatique et d'Analyse des Systèmes Rabat	Rapporteur
El Habib NFAOUI	PES	Faculté des Sciences Dhar El Mehraz de Fès	Rapporteur
Youssef EL ALLIOUI	PH	Faculté Polydisciplinaire de Khouribga	Rapporteur
Aziza EL OUAAZIZI	PH	Faculté Polydisciplinaire de Taza	Examineur
Mohamed BENSLIMANE	PH	École Supérieure de Technologie de Fès	Examineur
Khalil MAALMI	PES	École Supérieure de Technologie de Fès	Examineur
Mostafa HARTI	PES	Faculté des Sciences Dhar El Mehraz de Fès	Directeurs de thèse
Mohammed BERRADA	PH	École Nationale des Sciences Appliquées de Fès	

Laboratoire d'accueil : Laboratoire d'Intelligence Artificielle, de Sciences de données et Systèmes Émergents -LIASSE-

Établissement : École Nationale des Sciences Appliquées Fès.

“The cure for boredom is curiosity. There is no cure for curiosity.”

Dorothy Parker

Déclaration droits d'auteur

Je déclare que cette thèse intitulée « Élaboration d'une technique d'apprentissage automatique pour la détection de fraude par carte de crédit : Application sur les données massives déséquilibrées. » a été composée par moi-même et contient des travaux originaux de ma propre exécution. Certains des travaux rapportés ont été réalisés en collaboration avec un certain nombre de co-auteurs dont les contributions sont reconnues dans les sections pertinentes.

Salma EL HAJJAMI

Remerciements

A l'issue de la rédaction du manuscrit, je suis convaincue que la thèse est loin d'être un travail solitaire. L'accomplissement de ce travail a été possible grâce à plusieurs personnes qui ont eu une influence positive à travers leurs contributions scientifiques ou morales. C'est avec plaisir et reconnaissance que je leur réserve ces lignes.

En premier lieu, je tiens à exprimer mes remerciements les plus sincères à mes directeurs de thèse Monsieur Mohammed BERRADA, Professeur à l'ENSA (Ecole Nationale de Sciences Appliquées) et Monsieur Mostafa HARTI, Professeur à la FSDM (Faculté des Sciences Dhar El Mahraz), pour m'avoir fait confiance et pour la qualité d'encadrement et leurs apports tant au niveau des connaissances qu'au niveau humain, mais également pour leurs encouragements, ainsi que leur soutien tout au long de la thèse.

J'adresse également mes sincères remerciements aux membres du jury qui ont accepté d'évaluer mon travail :

- Mme. Nour EL Houda CHAOUI, PES, ENSA, Université Sidi Mohamed Ben Abdellah, Présidente ;
- M. Ahmed ZELLOU, PES, ENSIAS, Université Mohammed V, Rapporteur ;
- M. El Habib NFAOUI, PES, FSDM, Université Sidi Mohamed Ben Abdellah, Rapporteur ;
- M. Youssouf EL ALLIOUI, PH, FP, Université Sultan Moulay Slimane, Rapporteur ;
- M. Khalil MAALMI, PES, EST, Université Sidi Mohamed Ben Abdellah, Examinateur ;
- Mme. Aziza EL OUAAZIZI, PH, FP, Université Sidi Mohamed Ben Abdellah, Examinatrice ;
- M. Mohamed BENSLIMANE, PH, EST, Université Sidi Mohamed Ben Abdellah, Examineur.

Je suis reconnaissante de l'intérêt qu'ils ont porté à mes travaux.

Je souhaiterais aussi exprimer ma grande gratitude à Monsieur MALKI Jamal et Monsieur Alain Bouju pour m'avoir invité au laboratoire L3i de l'université de La Rochelle pour un stage de recherche, m'avoir guidé sur plusieurs pistes qui m'ont permis de progresser énormément et d'ouvrir des nouvelles perspectives.

Je suis également reconnaissante envers toute l'équipe pédagogique de l'École Nationale des Sciences Appliquées de Fès et les intervenants professionnels responsables de la filière génie informatique pour la qualité de formation qui m'a appris les notions initiales et développées des technologies de l'information et de la communication.

Ma reconnaissance serait incomplète sans remercier la plus grande source de ma force, ma famille. Ma plus profonde gratitude va à mes adorables parents pour m'avoir tout donné, en particulier leur amour, leur soutien indéfectible et leurs encouragements chaleureux, sans eux, je n'aurais pas été aussi loin.

À mon frère, ma soeur, mes amis et à toute personne que je n'ai pas su citer ici et qui m'ont apporté de la joie, du support, de l'aide, quand j'en avais besoin, merci.

Résumé

À l'ère numérique d'aujourd'hui, les cartes de crédit sont utilisées dans pratiquement toutes les transactions. La facilité de paiement en ligne au cours de la dernière décennie, a ouvert une nouvelle passerelle pour les opportunités, les entreprises, l'abaissement des frontières géographiques, mais aussi l'augmentation des transactions frauduleuses. Alors que le pic du graphique du commerce électronique ne cesse d'atteindre de nouveaux sommets chaque année, il est également devenu la toile de fond des fraudeurs qui abusent fortement de la transparence des méthodes de paiement en ligne et des transactions par carte de crédit. Par conséquent, la détection de fraude est un aspect essentiel qui consiste à trouver des anomalies dans les transactions. En général, la détection d'anomalies (AD) est le problème de la distinction entre les instances de données normales avec des motifs ou signatures bien définis et ceux qui ne sont pas conformes aux profils attendus. Ainsi le comportement frauduleux peut se manifester par des modèles inhabituels dans les transactions financières.

Les progrès récents des approches d'intelligence artificielle, comme l'apprentissage automatique ou Machine Learning (ML), font une percée importante dans ce domaine grâce à leur travail en temps réel, leur vitesse et leur efficacité. La détection de fraude qui était autrefois un domaine qui nécessitait beaucoup de travail manuel, est maintenant l'un des nombreux à avoir connu les progrès grâce aux algorithmes de ML. Néanmoins, la conception d'algorithmes de détection de fraude est sujette à divers problèmes, qui ont un impact considérable sur la précision de détection de fraude.

Le travail de recherche présenté dans cette thèse concerne l'élaboration d'une approche à base d'apprentissage automatique pour la détection de fraude. Nous traitons un problème majeur rencontré par les méthodes ML qui concerne le déséquilibre de classes, qui se produit lorsque l'une des classes appelée classe minoritaire est fortement sous-représentée par rapport à l'autre classe appelée classe majoritaire, et c'est particulièrement le cas du domaine de détection de fraude. En effet, la fraude est un événement rare par rapport au volume de transactions légitimes. L'ordre de grandeur est habituellement d'une transaction frauduleuse sur mille, ce qui perturbe les algorithmes ML qui présentent une grande précision sur les transactions normales mais une faible précision sur les transactions frauduleuses. Ce problème est plus vulnérable lorsque l'on considère le contexte des données massives.

Afin d'apporter des éléments de réponse pour cette problématique, nous proposons différentes contributions. Nous avons fourni comme première contribution, une étude expérimentale de l'influence du déséquilibre de classes sur les performances de prédiction des algorithmes d'apprentissage automatique pour comprendre les véritables faiblesses. Nous avons montré l'impact du déséquilibre des données sur la performance de ces approches.

Dans une seconde contribution, nous avons proposé une méthode appelée :« One Side Behavioral Noise Reduction (OSBNR) », qui agit au niveau des données d'apprentissage pour traiter le problème de déséquilibre de classes en présence d'un certain degré de bruit comportemental. OSBNR comprend deux principales étapes. Tout d'abord, une analyse de cluster est appliquée à des instances de la classe minoritaire, qui contient les transactions frauduleuses, pour construire plusieurs clusters de comportement similaire. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire, qui contient les transactions normales, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de comportement de la classe minoritaire.

Enfin, nous avons proposé une méthodologie générale basée sur la formalisation, l'implémentation et la conception technique d'une plateforme de détection de fraude en mesure de satisfaire les conditions de travail dans le monde réel. La partie formalisation se compose de quatre couches : le stockage distribué, le traitement des données, la construction du modèle et enfin l'évaluation du modèle. La partie implémentation utilise le système de traitement de données distribué Spark. En particulier, nous nous sommes basés sur sa bibliothèque dédiée au ML, appelé MLlib. La partie conception technique de la plateforme s'appuie sur des solutions open-source issues du monde du Big Data telles que Spark, Hadoop et Yarn.

Mots-clés: Détection de fraude, Détection d'anomalie, Apprentissage automatique, Déséquilibre de classes, Spark, MLlib, Big Data, Spark et Hadoop

Abstract

In today's digital age, credit cards are used in virtually all transactions. The ease of online payment over the past decade has opened a new door to opportunities, businesses, lowering geographic boundaries, but also in increase in fraudulent transactions. Fraudulent transactions can be made through various attempts such as theft of debit and credit cards, misuse of other cards and online accounts, and username and password hacking, etc. As the peak of the e-commerce graph continues to hit new highs every year, it has also become the backdrop for scammers who heavily abuse the transparency of online payment methods and credit card transactions. Therefore, fraud detection is an essential aspect of identifying fraudulent transactions that have been illegally issued on behalf of the legitimate cardholder and correcting any losses. Fraud detection is the challenge of finding anomalies in transaction data. In general, anomaly detection (AD) is the problem of distinguishing between normal data instances with well-defined patterns or signatures and those that do not conform to expected profiles. Fraudulent behavior can manifest itself in unusual patterns in financial transactions.

Recent advances in artificial intelligence approaches such as machine learning or Machine Learning (ML) are making an important breakthrough in this field thanks to their real-time work, speed and efficiency. Fraud detection which was once an area that required a lot of manual work is now one of many to have seen advancements in the ability of ML algorithms to learn historical fraud patterns and recognize them in futures transactions. However, the design of fraud detection algorithms is subject to various issues which have a huge impact on the accuracy of the system.

This thesis concern the development of machine learning-based technique for the detection of anomalies in banking transactions. We are addressing a major problem encountered by learning methods in fraud detection. This issue relates to the asymmetric distribution of classes, also known as class imbalance, which occurs when one of the classes has more instances than another class. This is particularly the case in the field of fraud detection, in fact we often encounter minority fraudulent transactions which are poorly represented during the learning phase. This problem confuses the learning algorithms which exhibit high precision on majority class instances and low precision on minority instances. This problem is more vulnerable when considering the context of big data.

In order to provide answers to this problem, we offer various contributions. As a first contribution, we provided an experimental study of the influence of class imbalance on the prediction performance of machine learning algorithms to understand the real weaknesses. We have shown the impact of the data imbalance on the performance of these approaches.

In a second contribution, we proposed a method that acts at the level of learning data called One Side Behavioral Noise Reduction (OSBNR). It is a sampling technique that

re-sample the data by under-sampling the majority class. OSBNR is based on two stages. First, a cluster analysis is applied to groups of similar instances of the minority class in multiple behavior clusters. Second, we select and eliminate instances of the majority class and we consider the behavioral noise which overlap with the behavior clusters of the minority class.

Finally, we have proposed a general methodology based on the formalization, implementation and technical design of a fraud detection platform capable of satisfying working conditions in the real world. The formalization part consists of four layers : distributed storage, data processing, model construction and finally model evaluation. The implementation part uses the Spark distributed data processing system. In particular, we base ourselves on its framework dedicated to ML, called MLlib. The technical design part of the platform is based on open-source solutions from the world of Big-Data such as Spark, Hadoop, Yarn.

Keywords: Fraud detection, Anomaly detection, Machine learning, Class imbalance, Big Data, Sampling method, Spark and Hadoop platforms

Environnement scientifique et notation

L’auteur a mené les recherches rapportées dans cette thèse aux Sciences de l’Ingénieur, Sciences Physiques, Mathématiques et Informatique, Université Sidi Mohamed Ben Abdellah, Fès, et au groupe de recherche au Laboratoire de l’Intelligence Artificielle, Sciences de données et Systèmes Emergents (IASSE), ENSA, Fès.

Notation

- x_i un vecteur défini dans un espace de description composée de plusieurs variables.
- y_i une valeur cible (ou label) associé à chaque vecteur x_i .
- n le nombre d’observations d’un ensemble de données.
- N_{maj} la taille de l’échantillon de la classe majoritaire.
- N_{min} la taille de l’échantillon de la classe minoritaire.
- $\mathcal{A}^+$ l’ensemble des données positives (minoritaires).
- $\mathcal{A}^-$ l’ensembles données négatives (majoritaires).
- $\mathcal{A}_0 = \{x_i \in \mathbb{R}^n, i = 1, \dots, n\}$ l’ensemble d’apprentissage.
- N^+ le nombre de données positives.
- N^- est le nombre de données négatives.

Terminologie

- Classifieur et Modèle : Sauf indication contraire, un classifieur fait référence à l’ensemble d’algorithmes utilisés pour estimer les paramètres à partir de données, effectuer des inférences sur des variables et convertir les valeurs inférées en classifications sur les enregistrements de données. Dans la littérature sur l’apprentissage automatique, le terme modèle est considérablement surchargé. Nous utilisons le terme modèle pour désigner la représentation abstraite des données que nous obtenons après avoir estimé des paramètres avec un classifieur sous une configuration d’hyperparamètre spécifique.
- Déséquilibre de classe : Distribution asymétrique des données où une classe est dominante et présente beaucoup plus que l’autre classe.
- Enregistrement de données : nous désignons un seul enregistrement de données, par ex. une transaction, génériquement comme enregistrement, observation, instance ou exemple.
- False Negative ou Faux négatif : Le nombre de positifs réels classés à tort comme négatifs.

- False Positive ou Faux positif : Le nombre de négatifs réels classés à tort comme positifs.
- Fraude : Une tromperie criminelle délibérément pratiquée par une personne afin de réaliser un profit financier illégalement.
- Ratio de déséquilibre : est défini comme le nombre d'exemples de classe négative divisé par le nombre d'exemples de classe positifs.
- Négative : La classe majoritaire (cas légitimes, authentiques ou non frauduleux).
- Positive : La classe minoritaire (observations de fraude, ou cas frauduleux).
- True Negative ou Vrai négatif : Le nombre d'observations correctement classées de la classe majoritaire.
- True Positive ou Vrai positif : Le nombre d'observations correctement classées de la classe minoritaire.

Liste des acronymes

- **Acc-RF-Sup** Accuracy - Relevant Features - Supervised Methods.
- **Acc-RF-Uns** Accuracy - Relevant Features - Unsupervised Methods.
- **Au-RF-Sup** AUC - Relevant Features - Supervised Methods.
- **Au-RF-Uns** AUC - Relevant Features - Unsupervised Methods.
- **AD** Anomaly Detection "Détection d'anomalies".
- **AE** Auto Encoder "Auto-encodeur".
- **All-KNN** All K-Nearest Neighbor "Tout k-plus proche voisin".
- **ANN** Artificial neural network "Réseau neuronal artificiel".
- **AUC** Area Under the receiver operating characteristic Curve "Aire sous la courbe de la fonction d'efficacité du récepteur".
- **BN** Bayesian Network "Réseau bayésien".
- **CMT** Case Management Tool "Outil de gestion de cas".
- **CNN** Condensed Nearest Neighbor "Voisin le plus proche condensé".
- **CNNET** Convolutional Neural Network "Réseau neuronal convolutif".
- **DDM** Data Driven Model "Modèle basé sur les données".
- **DT** Decision Tree "Arbre de décision".
- **EDR** Expert Driven Rules "Modèle piloté par des experts".
- **EM** Expectation Maximization "Maximisation des attentes".
- **ENN** Edited Nearest Neighbor "Voisin le plus proche modifié".
- **FDS** Fraud Detection System "Système de détection de fraude".
- **FPR** False Positive Rate "Taux de faux positifs".
- **FNR** False Negative Rate "Taux de faux négatifs".
- **GMM** Gaussian Mixture Model "Modèle de mélange gaussien".
- **HMM** Hidden Markov Model "Modèle de Markov caché".
- **ID3** Induction Decision Tree "Arbre de décision d'induction".
- **IF** Isolation Forest "Forêt d'isolement".
- **IR** Imbalance Ratio "Ratio de déséquilibre".
- **KDD** Knowledge Discovery in Databases "Découverte des connaissances dans les bases de données".
- **KNN** K-Nearest Neighbours "K-Voisins les plus proches".
- **LOF** Local Outlier Factor "Facteur de valeur aberrante locale".
- **LR** Logistic Regression "Régression logistique".
- **ML** Machine Learning "Apprentissage automatique".
- **MLP** Multi-Layer Perceptron "Perceptron multicouche".

- **NB** Naive Bayes.
- **NCR** Neighborhood Cleaning Rule "Règle de nettoyage avec voisinage".
- **NNET** Neural Network "Réseau neuronal".
- **OSBNR** One Side Noisy Behavioral Reduction "Réduction unilatérale du comportement bruyant".
- **OSS** One-Side Selection "sélection unilatérale".
- **PCA** Principal Component Analysis "Analyse en Composantes Principales".
- **RENN** Repeated Edited Nearest Neighbor "Voisin le plus proche modifié répété".
- **RF** Random Forest "Forêts aléatoires".
- **ROS** Random Over-Sampling "Sur-échantillonnage aléatoire".
- **RUS** Random Under-Sampling "Sous-échantillonnage aléatoire".
- **SAF** Supervised methods applied on All Features "Méthodes supervisées appliquées à toutes les fonctionnalités".
- **SMOTE** Synthetic Minority Oversampling Technique "Technique de suréchantillonnage des minorités synthétiques".
- **SOM** Self-Organizing Map "Carte auto-organisée".
- **SVM** Support Vector Machine "Machine à vecteurs de support".
- **TL** Tomek Link "Lien de Tomek".
- **TNR** True Negative Rate "Taux de vrais négatifs".
- **TPR** True Positive Rate "Taux de vrais positifs".
- **USAF** UnSupervised methods applied on All Features "Méthodes non supervisées appliquées à toutes les fonctionnalités".

Table des matières

Liste des tableaux	6
--------------------	---

Liste des figures	7
-------------------	---

Chapitre 1	
Introduction générale	10

1.1	Problèmes et motivations	11
1.2	Contributions	13
1.2.1	Contributions principales	13
1.2.2	Contributions secondaires	14
1.3	Délimitations	15
1.4	Portée de la recherche	15
1.5	Organisation du mémoire	16
1.6	Liste des publications et Communications	17

I Contexte général et état de l’art	18
--	-----------

Chapitre 2	
Contexte général et concepts fondamentaux	20

2.1	Introduction	21
2.2	La science des données	22
2.2.1	Qu’est-ce que la science des données ?	22

2.2.2	Comment considérer la science des données dans nos travaux? .	22
2.3	Apprentissage Automatique	23
2.3.1	Apprentissage automatique supervisé	24
2.3.2	Apprentissage non supervisé	24
2.4	Problématique de détection d'anomalies	25
2.4.1	Type d'anomalies	26
2.4.2	Les étiquettes de données	29
2.4.3	La sortie de la détection d'anomalies	29
2.5	Domaines d'application de la détection d'anomalies	30
2.6	Détection d'anomalie appliquée à la détection de fraude	31
2.6.1	Définition de la détection de fraude	31
2.6.2	Caractéristiques de la fraude	31
2.6.3	Système de détection de fraude du monde réel	32
2.7	Conclusion	35

Chapitre 3

État de l'art

36

3.1	Introduction	37
3.2	Mesures de performance des modèles d'apprentissage automatique . . .	38
3.3	Approches basées sur l'apprentissage automatique pour la détection des fraudes	42
3.3.1	Approches supervisées	42
3.3.2	Approches non supervisées	46
3.4	Apprentissage en distribution déséquilibrée	51
3.4.1	Distribution déséquilibrée : méthodes au niveau des données . .	52
3.4.2	Distribution déséquilibrée : méthodes au niveau de l'algorithme	56
3.5	Conclusion	61

II Contributions

63

Chapitre 4

Étude de performance des techniques d'apprentissage automatique dans une distribution déséquilibrée des données. 65

4.1	Introduction	66
4.2	Apprentissage automatique pour la détection d'anomalies	67
4.2.1	Contexte et problématique	67
4.2.2	Approche de détection d'anomalies supervisée	68
4.2.3	Approche de détection d'anomalies non supervisée	72
4.2.4	Approche générale	73
4.3	Évaluation expérimentale de performances de détection	74
4.3.1	Description de données	74
4.3.2	Protocole Expérimental	77
4.3.3	Analyse des Résultats	79
4.4	Conclusion	86

Chapitre 5

Une approche basée sur l'apprentissage automatique pour traiter le problème de déséquilibre de classes en présence de bruit comportemental. 87

5.1	Introduction	88
5.2	Le principe de notre approche : OSBNR	89
5.2.1	Contexte et problématique	89
5.2.2	Clustering de données	91
5.2.3	Notre Contribution : OSBNR	93
5.3	Étude expérimentale	97
5.3.1	Protocole expérimental	97

5.3.2	Analyse des résultats	98
5.4	Conclusion	110

Chapitre 6

Système de détection de fraudes basé sur l'apprentissage automatique.

Une approche méthodologique pour un système en production. 111

6.1	Introduction	112
6.2	Verrous liées aux données dans le système de detection d'anomalie . . .	114
6.3	Méthodologie d'apprentissage automatique et application à la détection des fraudes	115
6.3.1	Collecte et stockage des données	115
6.3.2	Le pré-traitement des données	115
6.3.3	Construction du modèle	116
6.3.4	Évaluation du modèle	117
6.4	Credit card fraud detection use case : Experimentation and Evaluation	118
6.4.1	Collecte des données	118
6.4.2	Le prétraitement des données	118
6.4.3	Construction du modèle	119
6.4.4	Évaluation du modèle	119
6.4.5	Analyse des résultats	120
6.5	Plateforme de développement de modèles ML	121
6.5.1	Architecture générale	121
6.5.2	Système distribuée de données	122
6.5.3	Système distribuée de traitements	122
6.5.4	Algorithmes de Spark ML	123
6.5.5	Environnement de développement intégré	124
6.6	Conclusion	125

Conclusion et perspectives **127**

Bibliographie	133
----------------------	------------

Annexe A	143
-----------------	------------

Annexe B	144
-----------------	------------

Annexe C	145
-----------------	------------

1	Environnement distribué Spark	145
1.1	Apache Hadoop	145
1.2	Apache Spark	147

Liste des tableaux

3.1	Matrice de confusion	38
3.2	Résumé des algorithmes d'apprentissage automatique pour la détection des fraudes	50
3.3	Résumé des approches de traitement de données déséquilibrées	59
3.3	Résumé des approches de traitement de données déséquilibrées	60
4.1	Détails de l'ensemble de données de fraude par carte de crédit Kaggle . . .	75
4.2	Statistiques des ensembles de données de formation et de test	78
5.1	Paramètres RF et MLP utilisés	98
5.2	Ensembles de formation et de test utilisés pour l'expérimentation	98
6.1	Paramètres RF et MLP utilisés	120
1	Versions utilisées	143

Liste des figures

2.1	Apprentissage supervisé.	24
2.2	Apprentissage non supervisé.	25
2.3	Partitionnement des données, ou clustering.	25
2.4	Un simple exemple d'anomalies dans un ensemble de données bidimensionnelles [21]	26
2.5	Composants clés associés à une technique de détection d'anomalies [21] . .	27
2.6	Exemple d'anomalie contextuelle [21]	28
2.7	Exemple d'anomalie collective [21]	28
2.8	Les couches d'un FDS([30]).	33
3.1	Une courbe ROC de base montrant les points importants [110]	41
3.2	Illustration de certaines stratégies de sous-échantillonnage	54
3.3	Illustration de certaines stratégies de sur-échantillonnage	55
4.1	Approche générale considérée	74
4.2	La distribution de la variable Time pour les deux classes	76
4.3	La distribution de la variable Amount pour les deux classes	76
4.4	Matrice de corrélation des variables de l'ensemble de données de carte de crédit	77
4.5	Mode de cluster Hadoop Yarn pour Spark	78
4.6	Étude de cas générale : toutes les métriques et toutes les fonctionnalités . .	80
4.7	SAF-78-22	81
4.8	USAF-78-22	81
4.9	Sélection des fonctionnalités pertinentes	82
4.10	Eval-General-4	83
4.11	Acc-RF-Sup	83
4.12	Au-RF-Sup	84

4.13	Eval-General-5	84
4.14	Acc-RF-Uns	85
4.15	Au-RF-Uns	85
5.1	Ensemble de données déséquilibré avec bruit comportemental	89
5.2	Clustering sur un ensemble de points 2D, 5 clusters	92
5.3	Illustration de l'algorithme K-Means. (a) Données d'entrée bidimensionnelles avec trois groupes; (b) trois points de départ sélectionnés comme centres de cluster et attribution initiale des points de données aux clusters; (c) et (d) des itérations intermédiaires mettant à jour les étiquettes de cluster et leurs centres; (e) regroupement final obtenu par l'algorithme K-Means à la convergence.	94
5.4	Flowchart de l'approche OSBNR	95
5.5	AUC pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalisés	99
5.6	AUC pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalisés	100
5.7	Précision pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalisés	101
5.8	Précision pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalisés	101
5.9	Rappel pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalisés	102
5.10	Rappel pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalisés	102
5.11	F1-score pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalisés	103
5.12	F1-score pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalisés	103
5.13	AUC pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base et cas des fonctionnalités pertinentes	104
5.14	AUC pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes	105
5.15	Précision pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base, et cas des fonctionnalités pertinentes	106
5.16	Précision pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes	106
5.17	Recall pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base, et cas des fonctionnalités pertinentes	107

5.18	Recall pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes . . .	107
5.19	F1-score pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base, et cas des fonctionnalités pertinentes . . .	108
5.20	F1-score pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes . . .	108
5.21	Performance des méthodes de ré-échantillonnage : en terme du temps de traitement	109
6.1	Valeur des transactions non monétaire dans le monde de 2017 à 2022 . . .	112
6.2	Flux de prétraitement des données	116
6.3	Flux de phase de formation	117
6.4	Flux de phase d'évaluation	118
6.5	Temps d'exécution total	121
1	exemple de l'implémentation de l'approche OSBNR en Scala	144
1	Ecosystème Apache Hadoop.	145
2	Ecosystème Apache Spark.	147

Chapitre 1

Introduction générale

Sommaire

1.1	Problèmes et motivations	11
1.2	Contributions	13
1.2.1	Contributions principales	13
1.2.2	Contributions secondaires	14
1.3	Délimitations	15
1.4	Portée de la recherche	15
1.5	Organisation du mémoire	16
1.6	Liste des publications et Communications	17

1.1 Problèmes et motivations

La fraude est aussi ancienne que l'humanité elle-même et peut prendre une variété illimitée de formes différentes. De plus, le développement de nouvelles technologies fournit des moyens supplémentaires permettant aux criminels de commettre une fraude, par exemple dans l'e-commerce, les informations sur la carte sont suffisantes pour commettre une fraude. L'utilisation des cartes de crédit est répandue dans la société moderne et la fraude par carte de crédit n'a cessé de croître ces dernières années. Les pertes financières dues à la fraude affectent non seulement les commerçants et les banques (par exemple les remboursements), mais également les clients individuels. Si la banque perd de l'argent, les clients finissent par payer également via des taux d'intérêt plus élevés, des frais d'adhésion plus élevés, etc. La fraude peut également affecter la réputation et l'image d'un commerçant causant des pertes non financières qui, bien que difficiles à quantifier à court terme, peuvent devenir visible dans la longue période. Par exemple, si un titulaire de carte est victime d'une fraude avec une certaine entreprise, il peut ne plus faire confiance à leur entreprise et choisir un concurrent.

Différents défis sont les facteurs moteurs de cette recherche. L'impact financier des activités frauduleuses est pour la plupart catastrophique. Un grand nombre d'organisations et d'individus sont quotidiennement victimes de fraudeurs. Malheureusement, cette tendance augmente chaque année. Ceci, par extension, a un impact socioculturel.

Les activités frauduleuses ont des conséquences financières désastreuses. Rien qu'en 2018, plus de 24 milliards de dollars a été perdu en raison de ces cas de fraude financière. Selon le rapport Nilson¹, les pertes de cartes mondiales devraient dépasser 34,66 milliards de dollars d'ici 2022. De 2015 à 2020, la fraude par carte dans le monde devrait totaliser 183,29 milliards de dollars [111].

De même, la fraude par carte de crédit était la forme la plus courante de vol d'identité en 2017, selon le FTC [52]. Elle a été classée numéro 1 des fraudes par vol d'identité - représentant 35,4% de toutes les fraudes par vol d'identité en 2018 [28]. Selon l'enquête 2020 sur la fraude à l'identité de Javelin Strategy & Research, la fraude d'identité totale a atteint 16,9 milliards de dollars (USD) en 2019, contre un record de 14,4 millions en 2018 aux États-Unis d'Amérique. Il s'agit d'un signal d'alarme, qui obligera les institutions financières, les entreprises et le secteur des paiements à réévaluer la manière dont la fraude d'identité est gérée [76].

De plus, les activités frauduleuses se multiplient chaque jour. Une enquête mondiale a été menée par KPMG en 2018. L'enquête de KPMG a révélé que 61% des personnes interrogées ont indiqué que le volume total de la fraude externe avait augmenté et 59% que la valeur avait augmenté [73]. Il a également été constaté que la tendance de la plupart des types de fraude est à la hausse. Par conséquent, l'impact financier des activités

1. Le rapport Nilson est le fournisseur de nouvelles et d'informations le plus respecté sur l'industrie mondiale des paiements par carte et mobile.

frauduleuses est le facteur clé qui a motivé cette initiative de recherche, car le problème est au cœur des scénarios du monde réel.

La fraude est un problème de longue date pour les industries. Un large éventail d'approches et solutions est disponible, mais rien ne peut arrêter les activités frauduleuses et les fraudeurs. C'est l'autre facteur important qui a motivé cette recherche. Bien que les systèmes de détection de fraude (FDS) aient acquis un pouvoir significatif au cours des dernières années, en adoptant de puissantes méthodologies fondées sur des statistiques et apprentissage automatique et en analysant des quantités massives de données pour découvrir des modèles et des mécanismes de fraude, la fraude reste extrêmement difficile à détecter [6]. Il existe divers problèmes qui entravent ces FDS. Le volume et la vitesse élevés des transactions générées, les distributions de données et la tendance des distributions à varier dans le temps, jouent tous un rôle essentiel dans la définition de la structure d'un système efficace de détection de fraude par carte de crédit.

Le principal problème du FDS d'aujourd'hui est de savoir comment améliorer la précision de la détection des fraudes avec un nombre croissant de transactions effectuées par utilisateur par seconde. L'augmentation du nombre d'utilisateurs et des transactions en ligne a entraîné de lourdes charges de travail sur ces systèmes. La taille des transactions quotidiennes et des transactions historiques passées a augmenté à grande échelle ces dernières années. Dans ce cas, le processus d'exploration et de détection des données sur les transactions entrantes avec un délai minimum est très difficile et compliqué à réaliser. En outre, cette situation ralentit également le processus de détection.

Un problème critique dans la détection de fraude par carte de crédit est la rareté des vrais ensembles de données provenant de l'industrie. Les données de paiement contiennent des informations privées très sensibles sur les individus et les entreprises et l'accès à ces données est strictement limité aux seuls propriétaires des données et aux entreprises qui gèrent les données. C'est la raison pour laquelle il n'existe pas d'un ensemble de données accessible au public qui refléterait suffisamment l'ampleur et la variété des paiements par carte dans le monde réel et qui pourrait donc être considéré comme une base pour étudier les nombreux défis intéressants présents dans ce domaine et évaluer les techniques existantes. Cependant, récemment, de nombreuses entreprises ont commencé à publier des extraits de leurs données entièrement anonymisées via un format de compétition (par exemple, Kaggle²), ce qui les rend sans valeur d'un point de vue commercial mais inestimable pour la communauté de la science des données.

Un autre problème concerne la distribution asymétrique (données déséquilibrées). Il est considéré comme l'un des problèmes les plus critiques rencontrés par les systèmes de détection de fraude. En général, le problème de données déséquilibrées est la situation où il y a beaucoup moins d'instances frauduleuses que d'instances normales [85]. Dans presque tous les ensembles de données, les transactions frauduleuses étaient nettement inférieures aux transactions légitimes, représentant environ 1 à 2% du nombre total d'instances. Les algorithmes utilisés pour la détection de la fraude par carte de crédit ont tendance à produire des classifieurs insatisfaisants lorsqu'ils sont confrontés à des ensembles de données déséquilibrés et chevauchés.

2. Plateforme la plus populaire pour les concours de science des données

De plus, les modèles dynamiques de fraude est un autre problème complexe. Les systèmes de détection de fraude ne peuvent détecter que les modèles de fraude sur la base de l'ensemble de données d'apprentissage qui se compose de modèles survenus dans le passé. Cependant, les fraudeurs changent constamment de tactique et certaines stratégies de fraude deviennent obsolètes, car le risque de détection a augmenté ou parce que l'efficacité de la méthode a chuté en raison de changements de sécurité. D'autres stratégies de fraude peuvent émerger des progrès technologiques ou des changements banals des habitudes des clients. Tout comme les changements de comportement frauduleux affectent les modèles de fraude, les changements dans le mode de vie des clients affectent leurs habitudes de dépenses. Le mode de vie lui-même peut présenter diverses notions de périodicité et varier en intensité dans le temps, influencé à la fois par les habitudes personnelles et les tendances générales (inflation, stabilité politique, etc.).

D'après la discussion ci-dessus, il est évident que les systèmes actuels de détection des fraudes présentent des lacunes qui permettent aux fraudeurs de poursuivre leurs actes criminels. En outre, ces activités doivent être arrêtées ou réduites pour diminuer l'impact de la fraude - qui nécessite une solution efficace. Dans cette thèse, nous étudions la détection de fraude basée sur les données transactionnelles et abordons plusieurs de ces défis complexes en utilisant des méthodes d'apprentissage automatique visant à identifier les transactions frauduleuses qui ont été émises illégitimement au nom du titulaire légitime de la carte.

1.2 Contributions

L'objectif de cette thèse est d'analyser et de développer des méthodes basées sur l'apprentissage automatique qui peuvent être utilisées pour détecter les transactions suspectes et potentiellement les intégrer dans un FDS. Il y a différentes contributions développées dans cette thèse et sont classées en contribution primaire et secondaire. Nous présentons ces contributions dans cette section.

1.2.1 Contributions principales

Dans cette thèse, des contributions principales peuvent être résumées comme suit :

- OSBNR : nous avons présenté une nouvelle approche appelée : One Side Behavioral Noise Reduction (OSBNR), de pré-traitement de données basée sur le clustering et le ré-échantillonnage afin de traiter le problème du déséquilibre et le chevauchement de classes. Notre approche est basée sur deux étapes. Premièrement, une analyse de clustering est appliquée pour grouper des instances similaires de la classe minoritaire dans plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de la classe minoritaire. Une série d'expériences a été réalisée pour valider la méthode proposée. Les résultats obtenus ont été comparés à plusieurs méthodes de ré-échantillonnage de pointe qui nous a

- permis de mieux évaluer notre approche qui s'est révélée prometteuse comme solution au problème du déséquilibre de classe. (Voir le chapitre 5 pour plus de détails.)
- Projet open-source spark-OSBNR : implémentation OSBNR en scala sous Spark, mis à la disposition de la communauté de recherche. Plus d'informations sur ce projet sont présentées à l'annexe B. Il s'agit d'une contribution utile, car les packages Spark pour travailler avec des données déséquilibrées sont rares.
 - Conception FDS : cette contribution couvre l'implémentation technique d'un FDS utilisant des technologies open-sources Hadoop et Spark. La conception, la mise en œuvre sont incluses ici. Notre implémentation dans Spark est hautement évolutive et fiable, ce qui est essentiel pour gérer la grande taille des ensembles de données du monde réel. Le processus d'obtention du meilleur modèle à mettre en œuvre a été décrit, Il contient la plupart des idées de conception communes dans les derniers FDSs, ce qui facilite grandement l'intégration des algorithmes de détection dans le flux de travail. Les résultats expérimentaux montrent que ce système a le potentiel d'atteindre une meilleure performance.(Voir le chapitre 6 pour plus de détails.)

1.2.2 Contributions secondaires

En plus des principales contributions discutées ci-dessus, nous avons mené une étude expérimentale et une étude de la littérature concernant les techniques ML pour détection de fraude et l'apprentissage dans un contexte de données déséquilibrées. Celles-ci sont essentiellement fortement liées à la contribution principale. Les études sont brièvement expliquées ci-dessous :

- Étude bibliographique : Une étude rigoureuse de la littérature a été menée principalement pour identifier les points forts et faibles des solutions existantes. L'étude a couvert les approches ML qui ont été adoptées pour la détection de la fraude par carte de crédit. nous avons discuté les deux grandes catégories d'algorithmes d'apprentissage avec leurs avantages et inconvénients. Deuxièmement, nous avons cité les différentes méthodes techniques proposées dans la littérature pour traiter le problème de données déséquilibrées, qui sont regroupées en deux grandes familles : les méthodes opérant au niveau de données et les méthodes opérant au niveau algorithmique. Enfin, nous avons introduit plusieurs métriques de performance bien adaptées dans ce contexte. Le chapitre 3 présente les résultats de l'enquête.
- Étude expérimentale : Nous avons évalué les modèles d'apprentissage supervisé et non supervisé pour la détection d'anomalies dans les transactions bancaires par carte de crédit en particulier. Nous avons mené une étude expérimentale en utilisant diverses approches supervisées et non supervisées. Plus précisément, 7 modèles de classification supervisés, à savoir, régression logistique (LR), Support Vector Machines (SVM), Decision Tree (DT), Random Forest (RF), Gradient-Boosted Tree (GBT), Naïve Bayes (NB) et Multilayer Perceptron (MLP) ainsi que 3 modèles de détection d'anomalies non supervisés, à savoir, K-Means (KM), Bisecting K-Means (BKM), et Gaussian Mixture Model (GMM), sont explorés dans cette étude. Nous avons évalué leurs performances sur un ensemble de données de transaction par

carte de crédit Kaggle³ en tenant compte de la distribution des anomalies dans un jeu de données déséquilibré. Cette étude expérimentale a été très efficace pour comprendre les véritables faiblesses des solutions existantes concernant le problème de déséquilibre de classe. Nous avons montré l'impact du déséquilibre des données sur la performance de ces approches. Ainsi, à partir des résultats obtenus, nous notons l'existence d'un ensemble de paramètres adaptés aux modèles supervisés et d'autres aux modèles non supervisés. Cette étude expérimentale est présentée au chapitre 4 .

1.3 Délimitations

Les principales délimitations de ce travail sont liées aux problèmes décrits précédemment :

- Comparaison des méthodes et techniques d'apprentissage : ce travail se concentre sur l'élaboration d'une technique d'apprentissage pour résoudre le problème de déséquilibre, et nous sommes intéressés par les méthodes ayant une implémentation officielle dans le cadre de calcul distribué open source Apache Spark.
- Limitations de Spark ML : le développement d'apprentissage automatique se fait à l'aide de Spark ML, donc tous les algorithmes et fonctionnalités non présents dans la version actuelle ne sont pas envisagés (les versions utilisées sont présentées dans l'annexe 6.6).

1.4 Portée de la recherche

La portée principale de cette recherche est triple : le traitement des données, l'analyse prédictive, et la visualisation des données. Ces portées sont brièvement présentées ci-dessous.

- Traitement des données : le traitement des données est une tâche courante et les tâches les plus critiques dans tout type d'analyse de données. Les techniques de traitement des données sont appliquées pour produire un ensemble de données de haute qualité qui est essentiel pour extraire des informations significatives des données. Cette recherche se concentre sur les techniques de nettoyage des données de fraude, de pré-traitement, de détection des valeurs aberrantes et de sur-échantillonnage ou de sous-échantillonnage. L'objectif était d'explorer les techniques existantes et d'identifier les plus adaptées au traitement des données financières.
- Analyse des données : c'est l'objet principal de cette thèse. L'objectif principal était de découvrir, concevoir et développer une technique de prédiction et de classification des observations. Les algorithmes d'apprentissage automatique ont été largement étudiés et utilisés pour atteindre l'objectif de cette thèse.

3. Plateforme la plus populaire pour les compétitions de science des données

- Visualisation des données : La visualisation des données comporte deux aspects : (i) les données sont visualisées pendant l'étude descriptive ou exploratoire pour avoir une représentation visuelle complète et une meilleure compréhension des données. En outre, les résultats finaux sont visualisés. Dans cette recherche, nous avons exploré les modèles de visualisation et utilisé les plus appropriés pour visualiser les données et les résultats.

1.5 Organisation du mémoire

Après avoir présenté notre motivation de cette thèse et notre problème de recherche, la suite de ce mémoire s'organise en deux parties : une première, composée des chapitres 2 et 3, elle a pour vocation d'approfondir le contexte de nos travaux ainsi que de l'état de l'art associé. La deuxième composée les chapitres 4, 5 et 6 présentera nos contributions avant une conclusion.

Partie 1

Dans cette partie, nous abordons le contexte, les concepts nécessaires et les travaux connexes.

- Chapitre 2 décrit le contexte général, en particulier, les notions de données depuis la science des données jusqu'aux dimensions de la qualité des données, qui conduisent à la découverte ou à la détection d'anomalies.
- Chapitre 3 présente les différents travaux dans l'état d'art sur les différentes techniques d'apprentissage automatique pour la détection de fraudes, et les différentes approches existantes pour traiter le problème de déséquilibre.

Partie 2

Dans cette partie, nous introduisons nos contributions.

- Chapitre 4 présente une évaluation expérimentale de performances des techniques d'apprentissage automatique pour identifier leurs faiblesses lorsqu'ils sont confrontés au problème de déséquilibre de classes, en comparant les deux catégories de techniques supervisées et non supervisées. Notre objectif est de détecter les problèmes qui doivent être résolus pour produire une solution hautement efficace pour le problème de déséquilibre de classe.
- Chapitre 5 introduit OSBNR (One Side Behavioral Noise Reduction), une approche de pré-traitement de données que nous avons développée, afin de traiter le problème du déséquilibre de classe. Notre méthode OSBNR (One Side Behavioral Noise Reduction) est basé sur deux étapes. Premièrement, une analyse de clustering est appliquée pour grouper des instances similaires de la classe minoritaire dans plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les

instances de la classe majoritaire, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de la classe minoritaire.

- Chapitre 6 présente une méthodologie générale basée sur la formalisation, la mise en œuvre et la conception technique d’une plateforme de détection de fraude. La partie formalisation se compose de quatre couches : le stockage distribué, le traitement des données, la construction du modèle et enfin l’évaluation du modèle. La partie implémentation utilise le système de traitement de données distribué Spark. En particulier, nous nous basons sur son cadre dédié au Machine Learning, appelé MLlib. La partie conception technique de la plateforme s’appuie sur les dernières technologies Big Data telles que Hadoop, Yarn, Livy etc.

Enfin, nous fournissons une conclusion générale pour le travail effectué dans cette thèse, et nous présentons les perspectives envisagées pour la suite d’autres recherches et travaux.

1.6 Liste des publications et Communications

Cette thèse a donné naissance aux publications et communications suivantes :

- EL HAJJAMI, Salma, MALKI, Jamal, BERRADA, Mohammed, and Fourka, Bouziane. Machine learning for anomaly detection. Performance study considering anomaly distribution in an imbalanced dataset. In IEEE, (Ed.) : The 5th International Conference on Cloud Computing and Artificial Intelligence : Technologies and Applications (CloudTech), Marrakesh, Morocco, IEEE Xplore Digital Library, 2020, ISBN : 978-1-7281-6175-4.
- EL HAJJAMI, Salma, MALKI, Jamal, BOUJU, Alain, and BERRADA, Mohammed. Machine Learning Facing Behavioral Noise Problem in an Imbalanced Data Using One Side Behavioral Noise Reduction : Application to a Fraud Detection. In World Academy of Science, Engineering and Technology International. Journal of Computer and Information Engineering, 14 (9), 2020.
- EL HAJJAMI, Salma, MALKI, Jamal, BERRADA, Mohammed, and Alain, Bouju. A Machine learning based approach to reduce behavioral noise problem in an imbalanced data. Application to a fraud detection. In : 2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA), Valencia, Spain, IEEE, 2020, pp. 11-20, doi : 10.1109/IDSTA50958.2020.9264114.
- El Hajjami, Salma, Malki, Jamal, Berrada, Mohammed, Harti, Mostafa, and Bouju, Alain, Machine learning system for fraud detection. A methodological approach for a development Platform, In : 2021 International Conference on Digital Technologies and Applications (ICDTA), Fez, Morocco, LNNS Springer Series.

Partie I

Contexte général et état de l'art

Table des matières

Chapitre 2	
Contexte général et concepts fondamentaux	
2.1	Introduction 21
2.2	La science des données 22
2.3	Apprentissage Automatique 23
2.4	Problématique de détection d'anomalies 25
2.5	Domaines d'application de la détection d'anomalies 30
2.6	Détection d'anomalie appliquée à la détection de fraude 31
2.7	Conclusion 35
Chapitre 3	
État de l'art	
3.1	Introduction 37
3.2	Mesures de performance des modèles d'apprentissage automatique . . . 38
3.3	Approches basées sur l'apprentissage automatique pour la détection des fraudes 42
3.4	Apprentissage en distribution déséquilibrée 51
3.5	Conclusion 61

Chapitre 2

Contexte général et concepts fondamentaux

Sommaire

2.1	Introduction	21
2.2	La science des données	22
2.2.1	Qu'est-ce que la science des données ?	22
2.2.2	Comment considérer la science des données dans nos travaux ?	22
2.3	Apprentissage Automatique	23
2.3.1	Apprentissage automatique supervisé	24
2.3.2	Apprentissage non supervisé	24
2.4	Problématique de détection d'anomalies	25
2.4.1	Type d'anomalies	26
2.4.2	Les étiquettes de données	29
2.4.3	La sortie de la détection d'anomalies	29
2.5	Domaines d'application de la détection d'anomalies	30
2.6	Détection d'anomalie appliquée à la détection de fraude	31
2.6.1	Définition de la détection de fraude	31
2.6.2	Caractéristiques de la fraude	31
2.6.3	Système de détection de fraude du monde réel	32
2.7	Conclusion	35

2.1 Introduction

Depuis quelques années, les données massives circulent dans le monde à travers des technologies nouvelles. Dans tous les domaines : e-commerce, e-tourisme, e-santé, e-éducation, e-administration, réseaux sociaux, etc. On le sait, ces données représentent un avantage considérable. Elles sont représentatives du bien-être des systèmes et de leurs applications. Par conséquent, tout écart par rapport au comportement, dit normal, est d'un intérêt particulier et peut nécessiter une action.

Pour faire face à cette problématique, les approches de prise en charge des données massives vont connaître un intérêt majeur. Plusieurs systèmes et solutions sont alors développés. Les entreprises et plusieurs autres organisations utilisent ces solutions et contribuent ainsi fortement à améliorer leurs efficacité. Cette nouvelle tendance, a transformé l'analyse des données en l'orientant vers certaines approches spécialisées sous l'égide de la science des données « Data Science en anglais ». En résumé, ce nouveau mouvement de la science des données peut être considérée comme une discipline permettant de découvrir des relations, des modèles, des tendances nouveaux et significatifs, lors de l'examen de grandes quantités de données.

La problématique de détection d'anomalies est une partie importante de la science des données. Elle est en plein essor et revêt une importance cruciale dans plusieurs domaines. En effet, cette discipline utilise les approches de la science des données pour détecter les données anormales ou aberrantes dans un ensemble de données [21]. Ainsi, la recherche et développement dans cette discipline tendent à construire des modèles, de plus en plus sophistiqués, pour étudier le comportement des données.

Pour comprendre et résoudre un problème de détection d'anomalies, nous devons avoir une bonne compréhension de certains concepts de base de la science des données pour la mise en place d'un modèle approprié. Ce chapitre fournit le contexte nécessaire à ce sujet, en particulier, les notions fondamentaux depuis la science des données jusqu'aux dimensions de la qualité des données, qui conduisent à la découverte ou à la détection d'anomalies. Ensuite, nous nous concentrons sur le cas de détection de fraude accompagné d'une brève description du système réel de détection de fraude.

Ce premier chapitre est structuré en cinq sections. La première section décrit d'abord le terme de la science des données, son flux de travail et les problèmes standards de la science des données. Dans la deuxième section, nous présentons des connaissances préliminaires sur l'apprentissage automatique qui est devenu la principale technique pour la détection d'anomalies et qui peut se faire suivant deux paradigmes connus apprentissage supervisé et apprentissage non supervisé. Dans la troisième section, nous introduisons le terme du Big Data ou données massives. Dans la quatrième section, des notions relatives à la détection d'anomalies sont présentés. Nous terminons ce chapitre par une présentation du problème de la détection des fraudes qui est le domaine d'application de notre thèse.

2.2 La science des données

2.2.1 Qu'est-ce que la science des données ?

La science des données peut être définie comme étant un ensemble de principes, de définitions de problèmes, d'algorithmes et de processus pour construire des modèles utiles à partir de données massives. De nombreux éléments de la science des données sont étudiés et développés dans des domaines adjacents tels que l'apprentissage automatique (Machine Learning (ML)) et l'exploration de données [109].

En réalité, les termes science des données, apprentissage automatique et exploration de données, et peut être d'autres, sont souvent utilisés de manière interchangeable. Toutes ces disciplines partagent un point commun important qui consiste à l'amélioration de la prise de décision grâce à l'analyse des données. Cependant, bien que la science des données emprunte beaucoup de choses à ces autres domaines, sa portée est plus large. L'apprentissage automatique (ML) se concentre sur la conception et l'évaluation d'algorithmes pour extraire des modèles à partir de données. L'exploration de données traite généralement de l'analyse de données, souvent en mode structuré, et implique souvent un accent sur les applications commerciales. La science des données tient compte de toutes ces considérations, mais relève également d'autres défis, tels que la capture, le nettoyage et la transformation de toutes sortes de données de toutes origines [109]. La science de données se distingue aussi par l'utilisation de technologies de mégadonnées, dites aussi grands ensembles de données, ou le Big-Data, pour stocker et traiter les données massives. La science de données s'intéresse aussi aux problématiques de gouvernance, les questions liées à l'éthique et à la réglementation des données.

2.2.2 Comment considérer la science des données dans nos travaux ?

Plusieurs travaux de recherche se sont penchés sur la définition des composantes structurelles ou les courants gouvernant la science de données [27]. Dans [13], nous trouvons une distinction entre deux cultures :

1. l'analyse statistique : dans sa forme classique et dans la lignée des grands penseurs de cette discipline (Gauss, Pearson et tant d'autres) ;
2. l'analyse algorithmique : modélisation algorithmique, tant en théorie qu'en pratique qui s'est développée rapidement dans des domaines autres que les statistiques. Elle peut être utilisée à la fois sur de grands ensembles de données complexes et comme une alternative plus précise et informative à la modélisation de données sur des ensembles de données plus petits.

Toujours selon Breiman [13], la culture classique se base sur la présomption que les données analysées sont générées par des processus stochastiques que l'on cherche à décrire. La culture algorithmique, s'intéresse aux processus sous-jacents aux données se concentrant à en extraire de l'information avec des modèles qui peuvent être des boîtes noires.

L'ensemble de nos travaux décrits dans ce rapport, qui relèvent tout naturellement de la science des données, vont suivre une approche basée sur la pensée de Breiman. Dans cette alignée, notre approche se distingue des autres car il définit les composantes structurelles dont elle a besoin pour fonctionner. Il s'agit des composantes suivantes :

1. Statistiques : elles ont un objectif similaire à l'analyse des données. Elles ont aussi la capacité de faire des inférences à partir de ces données, d'où le nom : « Statistiques Inférentielles [40] ».
2. Apprentissage automatique : l'apprentissage automatique est le domaine de l'intelligence artificielle consacré à étudier un postulat simple mais fondamental qui est : les systèmes doivent apprendre. Certaines méthodes de l'apprentissage automatique peuvent être utilisées pour construire des modèles à partir de données, c'est-à-dire pour construire des explications, et pour faire des prédictions dans des problèmes de classification et de régression. Le lien entre apprentissage automatique et statistique, en particulier inférentielle, est très important. Plusieurs approches et outils développés dans le cadre de l'apprentissage automatique sont étroitement liés ou se chevauchent avec des méthodes statistiques.
3. Big-Data et technologies des base de données : la taille d'une base de données n'est pas nécessairement un aspect crucial pour construire un modèle et prendre une décision. Toutefois, la pertinence de la science des données, dans tout système en production, réside dans le fait que les données sont le moteur de ces systèmes, et que des données massives sont stockées pour être analysées. Les modèles, comme ceux construits par une approche apprentissage automatique, doivent être implémentés. Cela signifie que des algorithmes sous forme de briques logicielles doivent être programmés efficacement et conduire à des solutions dans des délais raisonnables. Pour y parvenir, on doit alors maîtriser les technologies du Big Data.

Comme dit précédemment, il n'existe pas de consensus partagé sur les domaines que la science des données intègre. Dans d'autres travaux, on peut trouver une composante structurelle d'exploration de données à la place ou en conjonction avec l'apprentissage automatique.

Notre considération de la science des données en trois composantes structurelles a besoin de connaissances spécialisées. Ces dernières sont formulées et explicitées par un expert du domaine. Les trois composantes structurelles peuvent être fortement impactées par considérations inhérentes à des connaissances spécialisées. Par conséquent, nous avons aussi besoin d'un expert de domaine.

Dans la suite, nous allons aborder les deux composantes structurelles concernant l'apprentissage automatique et les technologies du Big-Data.

2.3 Apprentissage Automatique

L'apprentissage automatique concerne l'exploration et le développement de modèles mathématiques (statistiques et autres) en utilisant des algorithmes appropriés dans le but

d'apprendre à partir des données. Dans cette section, nous nous intéressons aux problèmes à deux classes dans le domaine de l'apprentissage automatique.

2.3.1 Apprentissage automatique supervisé

L'apprentissage automatique supervisé est peut-être le type de problèmes le plus facile à appréhender. En effet, son objectif est d'apprendre à faire des prédictions, à partir d'une liste d'instances étiquetées, c'est-à-dire accompagnés de la valeur à prédire (voir Figure 2.1). Les étiquettes servent de « professeur » et supervisent l'apprentissage de l'algorithme.

Dans [5], on représente la branche d'apprentissage automatique qui s'intéresse aux problèmes pouvant être formalisés de la façon suivante :

Définition 1 (Apprentissage supervisé) *Étant données n observations $x_i, i = 1, \dots, n$ décrites dans un espace $\mathcal{X}$, et leurs étiquettes $y_i, i = 1, \dots, n$ décrites dans un espace $\mathcal{Y}$, on suppose que les étiquettes peuvent être obtenues à partir des observations grâce à une fonction $\phi : \mathcal{X} \rightarrow \mathcal{Y}$ fixe et inconnue : $y_i = \phi(x_i) + \epsilon_i$, où ϵ_i est un bruit aléatoire. Il s'agit alors d'utiliser les données pour déterminer une fonction $f : \mathcal{X} \rightarrow \mathcal{Y}$ telle que, pour tout couple $(x, \phi(x)) \in \mathcal{X} \times \mathcal{Y}$, $f(x) \approx \phi(x)$.*

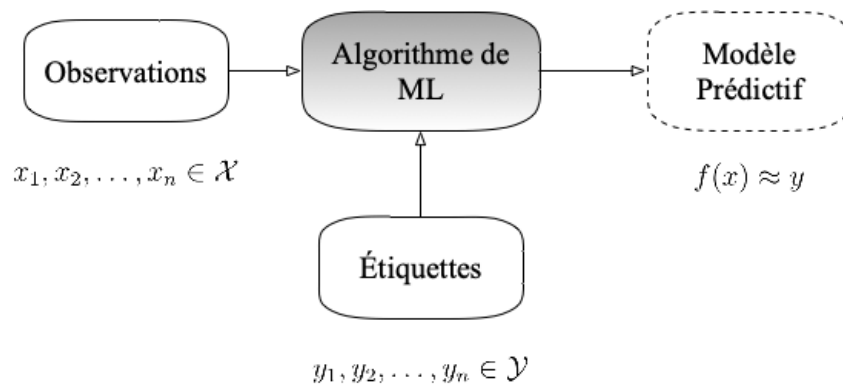


FIGURE 2.1 – Apprentissage supervisé.

L'espace dans lequel sont définies les données est le plus souvent $X = \mathbb{R}^p$. Il existe d'autres types de représentations des étiquettes, comme variables binaires, discrètes, catégoriques, voire des chaînes de caractères ou des graphes.

2.3.2 Apprentissage non supervisé

Dans le cadre de l'apprentissage non supervisé, les données ne sont pas étiquetées. Il s'agit alors de modéliser les observations pour mieux les comprendre (voir Figure 2.2).

Dans [5], on représente la branche de l'apprentissage automatique qui s'intéresse aux problèmes pouvant être formalisés de la façon suivante :

Définition 2 (Apprentissage non supervisé) *Étant données n observations $x_i, i = 1, \dots, n$ décrites dans un espace $\mathcal{X}$, il s'agit d'apprendre une fonction sur $\mathcal{X}$ qui vérifie certaines propriétés.*



FIGURE 2.2 – Apprentissage non supervisé.

Cette définition est très vague, et sera certainement plus claire sur des exemples. Tout d'abord, le regroupement, ou partitionnement, consiste à identifier des groupes dans les données (voir Figure 2.3). Cela permet de comprendre leurs caractéristiques générales, et éventuellement d'en déduire les propriétés d'une observation en fonction du groupe auquel elle appartient.

Définition 3 (Partitionnement) *Le partitionnement représente un problème d'apprentissage non supervisé pouvant être formalisé comme la recherche d'un groupe $\bigcup_{k=1}^K C_k$ des n observations $x_i, i = 1, \dots, n$. Ce groupe doit être pertinent au vu d'un ou plusieurs critères à préciser.*



FIGURE 2.3 – Partitionnement des données, ou clustering.

2.4 Problématique de détection d'anomalies

La détection des anomalies est avant tout un problème de la science des données qui a été étudié dans divers domaines de recherche et domaines d'application. La détection d'anomalies est un processus d'analyse de données qui détecte les données anormales, aberrantes, ou qui ne sont pas conformes au comportement attendu, dans un ensemble de données [21, 3]. Ces données non conformes sont souvent appelés anomalies, mais en fonction du domaine d'application, on peut trouver d'autres descriptions telles que des valeurs aberrantes, des observations discordantes, des exceptions, des aberrations, des surprises, des particularités ou des contaminants [58, 100, 3]. La détection d'anomalies a été largement étudiée dans les statistiques et l'apprentissage automatique, et a également été

décrite comme synonyme de détection de valeurs aberrantes, de détection de nouveauté, de détection d'écart et d'extraction d'exception. La Figure 2.4 illustre les anomalies dans un simple ensemble de données bidimensionnelles. Les données ont deux régions normales, N_1 et N_2 , puisque la plupart des observations se situent dans ces deux régions. Les points suffisamment éloignés de ces régions, par exemple les points o_1 et o_2 , et les points de la région O_3 , sont des anomalies.

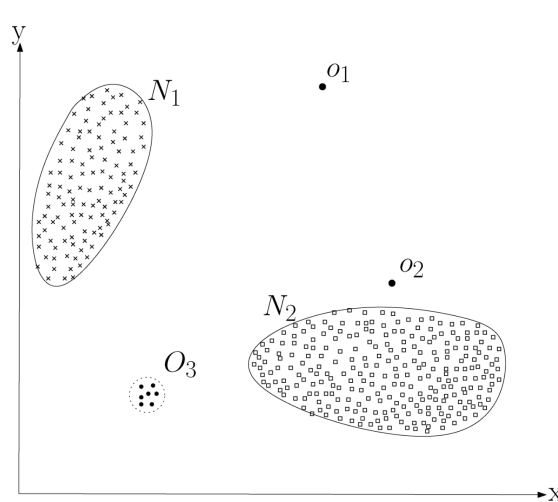


FIGURE 2.4 – Un simple exemple d'anomalies dans un ensemble de données bidimensionnelles [21]

En général, une anomalie est définie comme un point de données ou un ensemble de points de données sensiblement différent des valeurs historiques ou des valeurs attendues. Il existe différentes définitions de la différence entre les points de données, ce qui conduit à différents types d'anomalies. La détection des anomalies est considérée comme cruciale car elle indique des événements importants mais rares et peut déclencher des actions critiques. Il a été largement appliqué dans d'innombrables domaines d'application tels que le biomédical, la finance, la vidéosurveillance, la sécurité, les systèmes de réseaux et la détection de fraudes. Par exemple, des données anormales dans les transactions par carte de crédit pourraient indiquer des activités frauduleuses. Un schéma inhabituel d'un trafic réseaux peut signifier qu'un ordinateur a été piraté et que des données sont transmises à des destinations non autorisées. Une anomalie sur une image IRM (imagerie par résonance magnétique) pourrait indiquer la présence d'une tumeur maligne [3, 4]. La Figure 2.5 illustre les composants clés associés à une technique de détection d'anomalies.

2.4.1 Type d'anomalies

Un aspect important d'une technique de détection d'anomalie est la nature de l'anomalie elle-même. Dans la littérature [21, 3], les anomalies peuvent être classées en trois principaux types :

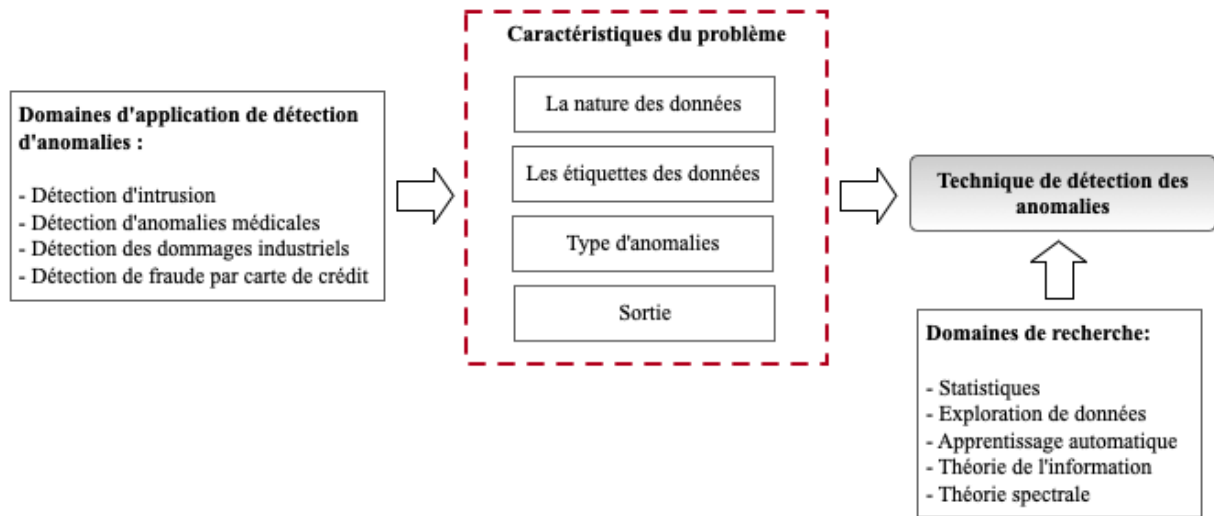


FIGURE 2.5 – Composants clés associés à une technique de détection d'anomalies [21]

- Anomalie ponctuelle : si une donnée individuelle peut être considérée comme anormale par rapport au reste des données, alors elle est appelée une anomalie ponctuelle. Il s'agit du type d'anomalie le plus simple et c'est l'objet de la majorité des recherches sur la détection des anomalies. Par exemple, une dépense de carte de crédit exceptionnelle par rapport aux transactions historiques est une anomalie ponctuelle. la Figure 2.4 montre un exemple d'anomalie ponctuelle. Les points o_1 et o_2 , ainsi que les points de la région O_3 , se trouvent en dehors de la limite des régions normales, et sont donc des anomalies ponctuelles car ils sont différents des points de données normaux.
- Anomalie contextuelle : dans ce type d'anomalie, on ne peut repérer le point de données anormal qu'en regardant le contexte auquel appartient cette observation. La notion de contexte est induite par la structure de l'ensemble de données et doit être spécifiée dans le cadre de la formulation du problème. Chaque donnée est définie à l'aide des deux ensembles d'attributs suivants :
 1. Attributs contextuels : utilisés pour déterminer le contexte (ou le voisinage) de cette donnée. Par exemple, dans les données de séries chronologiques, le temps est un attribut contextuel qui détermine la position d'une donnée sur toute la séquence.
 2. Attributs comportementaux : définissent les caractéristiques non contextuelles d'une donnée. Par exemple, dans un ensemble de données spatiales décrivant les précipitations moyennes du monde entier, la quantité de précipitations à n'importe quel endroit est un attribut comportemental.

La Figure 2.6 montre un tel exemple pour une série chronologique de température qui montre la température mensuelle d'une zone au cours des dernières 2.6 années. Une température de $35^\circ F$ pourrait être normale pendant l'hiver (au temps t_1) à cet endroit, mais la même valeur pendant l'été (au temps t_2) serait une anomalie.

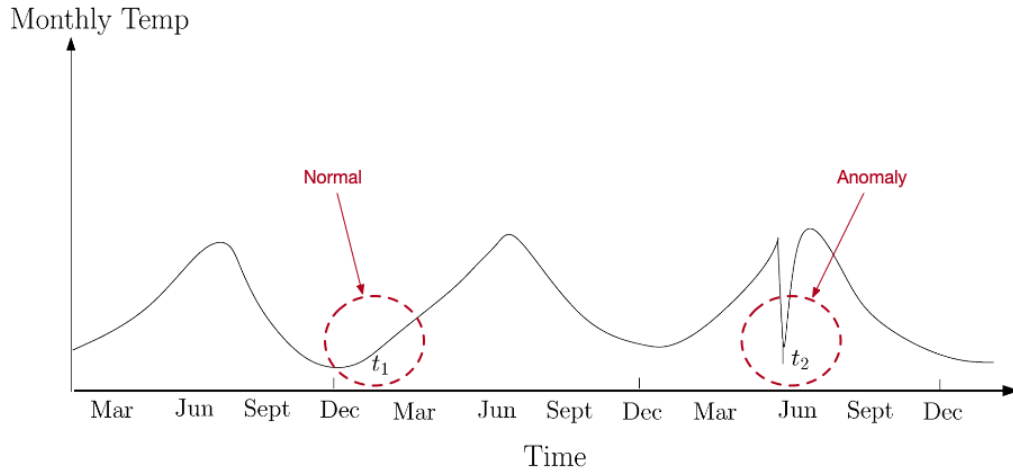


FIGURE 2.6 – Exemple d'anomalie contextuelle [21]

- Anomalie collective : si une collection de données est anormale par rapport à l'ensemble de données, on parle d'anomalie collective. Les données individuelles dans une anomalie collective peuvent ne pas être des anomalies en elles-mêmes, mais leur occurrence ensemble en tant que collection est anormale. La Figure 2.7 est un exemple qui montre une sortie d'électrocardiogramme humain [21]. La région encadrée dénote une anomalie car la même valeur basse existe pendant une durée anormalement longue, correspondant à une contraction auriculaire prématurée.

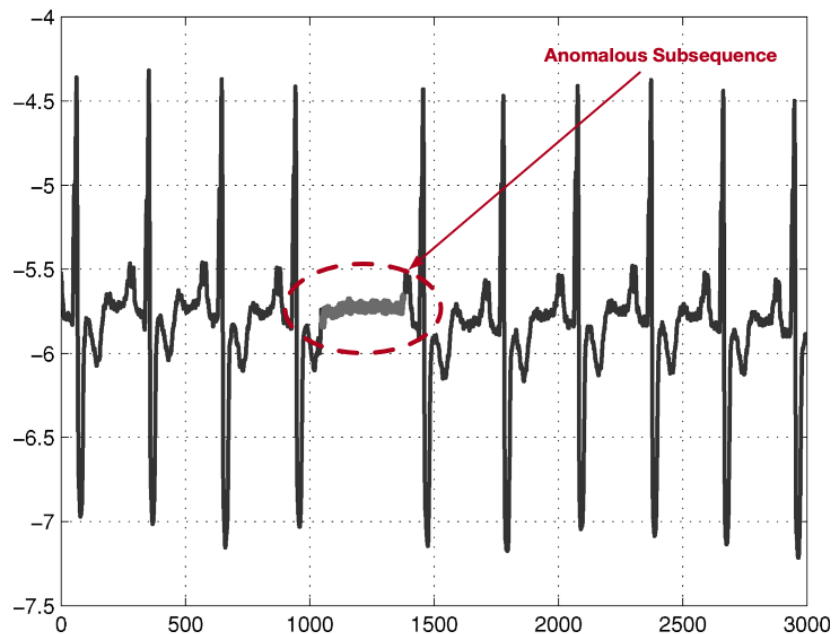


FIGURE 2.7 – Exemple d'anomalie collective [21]

2.4.2 Les étiquettes de données

Les étiquettes liées à un point de données indiquent si ce point est normal ou anormal [21]. On sait que l'obtention de données étiquetées exactes et caractéristiques de tous les types de comportements est souvent trop coûteuse. L'étiquetage est souvent effectué par un expert humain et, par conséquent, un effort considérable est nécessaire pour obtenir l'ensemble de données d'apprentissage étiqueté [21]. Normalement, obtenir un ensemble étiqueté de données anormales qui couvre tous les types possibles de comportement anormal est plus difficile que de recevoir des étiquettes pour un comportement normal [21].

En fonction de la portée des étiquettes disponibles, les techniques de détection d'anomalies peuvent fonctionner sur l'un des deux modes suivants :

1. Détection d'anomalies supervisées : en mode supervisé, les techniques entraînées supposent la disponibilité d'un ensemble de données d'apprentissage qui a des données étiquetées pour les classes normales et anormales ou anomalies. Une approche typique dans de tels cas consiste à construire un modèle prédictif pour les classes normales par rapport aux classes d'anomalies. Deux problèmes majeurs se posent lors de la détection d'anomalies supervisées. Premièrement, les instances anormales sont beaucoup moins nombreuses que les instances normales dans les données d'entraînement. Les problèmes qui surviennent en raison de la distribution de classes déséquilibrée ont été abordés dans la littérature [48, 119, 129, 125]. Deuxièmement, il est généralement difficile d'obtenir des étiquettes précises et représentatives, en particulier pour la classe d'anomalies.
2. Détection d'anomalies non supervisées : en mode non supervisé, les techniques ne nécessitent pas de données d'apprentissage et sont donc les plus largement applicables. Les techniques de cette catégorie supposent implicitement que les instances normales sont beaucoup plus fréquentes que les anomalies dans les données de test. Si cette hypothèse n'est pas vraie, ces techniques souffrent d'un taux élevé de fausses alarmes.

2.4.3 La sortie de la détection d'anomalies

Une caractéristique importante de toute technique de détection d'anomalies est la manière dont les anomalies sont décrites. Selon plusieurs travaux dont [21], en règle générale, les sorties produites par les techniques de détection d'anomalies sont de l'un des deux types suivants :

1. Scores : les techniques de notation attribuent un score d'anomalie à chaque point des données de test en fonction du degré auquel ce point est considéré comme une anomalie. Ainsi, le résultat de ces techniques est une liste classée d'anomalies. Un analyste peut choisir d'analyser les anomalies les plus élevées ou d'utiliser un seuil de coupure pour sélectionner les anomalies.
2. Étiquettes : dans cette catégorie, les techniques attribuent une étiquette « normal » ou « anormal » à chaque instance de test.

2.5 Domaines d'application de la détection d'anomalies

La détection d'anomalies est utilisée dans une variété de domaines, tels que la détection d'intrusion, la détection de fraude, la détection de défaut, la surveillance de l'état du système, la détection d'événements dans les réseaux de capteurs et la détection des perturbations de l'écosystème. Dans cette section, nous discutons quelques applications de la détection d'anomalies.

- Détection d'intrusion : fait référence à la détection d'activités malveillantes (effractions, pénétrations et autres formes d'abus informatique) dans un système informatique [80, 3]. Ces activités malveillantes ou intrusions sont intéressantes du point de vue de la sécurité informatique. Une intrusion est différente du comportement normal du système, et par conséquent, les techniques de détection d'anomalies sont applicables dans le domaine de détection d'intrusion.
- Détection des anomalies dans le domaine médical : fonctionne généralement avec les dossiers des patients. Les données peuvent présenter des anomalies pour plusieurs raisons, telle qu'un état anormal du patient, des erreurs d'instrumentation ou des erreurs d'enregistrement. Plusieurs techniques se sont également concentrées sur la détection des flambées de maladies dans une zone spécifique [Wong et al. 2003]. Ainsi, la détection d'anomalies est une problématique critique dans ce domaine et nécessite une grande précision [120, 45].
- Détection des dommages industriels : les unités industrielles subissent des dommages dus à une utilisation continue et à une usure normale. Ces dommages doivent être détectés tôt pour éviter une nouvelle escalade et des pertes. Les données de ce domaine sont généralement appelées « données de capteur » car elles sont enregistrées à l'aide de différents capteurs et collectées pour analyse. Des techniques de détection d'anomalies ont été largement appliquées dans ce domaine pour détecter de tels dommages [78]. La détection des dommages industriels peut être classée en deux domaines. Le premier traite des défauts dans les composants mécaniques tels que les moteurs, etc., et le second traite des défauts des structures physiques. Le premier domaine est également appelé gestion de l'intégrité du système.
- Détection de fraude : fait référence à la détection d'activités criminelles se déroulant dans des organisations commerciales telles que les banques, les sociétés de cartes de crédit, les agences d'assurance, les sociétés de téléphonie mobile, le marché boursier, etc. Les utilisateurs malveillants peuvent être les véritables clients de l'organisation ou se faire passer pour des clients (également appelé vol d'identité). La fraude se produit lorsque ces utilisateurs consomment les ressources fournies par l'organisation d'une manière non autorisée. Les organisations sont intéressées par la détection immédiate de ces fraudes pour éviter des pertes économiques [89, 4, 93].

2.6 Détection d'anomalie appliquée à la détection de fraude

2.6.1 Définition de la détection de fraude

Nous nous intéressons au problème de détection d'anomalie appliqué à la détection de fraude. Ainsi, nos travaux portent sur la détection de fraude dans un ensemble de transactions par carte de crédit. Il s'agit du processus d'identification si une nouvelle transaction autorisée appartient à la classe des transactions frauduleuses ou authentiques [85]. Nous parlons alors de Système de Détection de Fraudes (FDS). Un tel système devrait non seulement détecter efficacement les cas de fraudes, mais aussi être rentable en ce sens que le coût investi dans le filtrage des transactions ne devrait pas être supérieur à la perte due aux fraudes [95]. Il existe de nombreuses stratégies différentes opérées par les fraudeurs. Nous présentons brièvement les procédures les plus courantes dans les scénarios hors ligne et en ligne.

- **Fraude hors ligne** est un type rare de fraude où le fraudeur vole réellement la carte de crédit physique du titulaire de la carte ou copie la bande magnétique. Dans la plupart des cas, la victime contacte sa banque pour bloquer instantanément la carte de crédit perdue de sorte que les transactions effectuées par cette carte après son vol se voient refusées.
- **Fraude en ligne** est aujourd'hui la fraude typique dans laquelle un fraudeur vole les informations de la carte de crédit par des moyens en ligne malveillants. Ce type de fraude est le plus dangereux car le client n'est pas au courant du vol [89] et peut se produire à travers différentes stratégies. Parmi les stratégies les plus fréquentes, la première est l'écrouillage où l'idée est de voler les informations du titulaire de la carte lors d'une véritable transaction via un terminal de paiement modifié qui stocke toutes les informations de la carte. Ces dernières années, les distributeurs automatiques de billets ont été utilisés par les skimmers pour extraire les informations de la carte, c'est-à-dire en définissant un faux clavier, un appareil photo ou d'autres outils qui capturent les informations pertinentes de la carte. Une autre stratégie très populaire est appelée hameçonnage, où le fraudeur utilise un site Web pour voler les informations de la carte en clonant une carte existante ou simplement en créant une avec des processus de paiement non sécurisés. D'autres stratégies existent, comme la diffusion de logiciels malveillants de type cheval de Troie, qui est devenue une pratique très courante. Ce type de stratégie n'est cependant pas aussi efficace que les précédentes étant donné qu'il est difficile d'extraire les informations pertinentes sur la carte de crédit. Enfin, des astuces simples telles que tromper le titulaire de la carte avec des courriers malveillants sont également courantes.

2.6.2 Caractéristiques de la fraude

La fraude constitue un phénomène récurrent, ancien, mais toujours vivace et qui ne cesse de s'adapter à l'évolution de nos sociétés. Dans [115], les auteurs ont caractérisé le

phénomène multiforme de la fraude : crime rare, imperceptiblement dissimulé, évoluant dans le temps et souvent soigneusement organisé, qui se présente sous de nombreuses formes. Dans [6], ces caractéristiques sont résumées comme suit :

- La fraude est un événement rare : indépendamment du cadre ou de l'application exacte, seule une minorité de la population est concernée par la fraude. Donc, il y a moins d'instances frauduleuses que d'instances normales [85]. Par exemple, dans le cadre de la fraude par carte de crédit, généralement moins de 0,5% des transactions sont frauduleuses. Un tel problème est communément appelé problème d'aiguille dans une botte de foin. En outre, les fraudeurs masquent les activités frauduleuses avec les activités non frauduleuses, ce qui est une autre cause de fraude rare.
- La Fraude est un comportement réfléchi : une fois que les fraudeurs ont trouvé un moyen pour réaliser l'escroquerie, ils l'exploitent jusqu'à ce que ce type de fraude soit découvert et que des mesures de prévention soient prises.
- La Fraude est un comportement imperceptiblement dissimulé : le comportement des fraudeurs ne se distingue pas des autres. Ils gardent le comportement normal pour passer inaperçu et rester couverts par des non-fraudeurs. Cela rend effectivement la fraude imperceptiblement dissimulée, car les fraudeurs réussissent à se cacher en réfléchissant et en planifiant bien comment commettre une fraude avec précision.
- La Fraude évolue dans le temps : les fraudeurs changent continuellement de méthode parce que leur objectif est de ne pas être détecté autant que possible. Cela signifie que les techniques et astuces utilisées par les fraudeurs évoluent dans le temps en s'adaptant aux mécanismes de détection de fraude.
- La Fraude est généralement un crime soigneusement organisé : les fraudeurs organisent très soigneusement les activités frauduleuses comme les autres crimes organisés. Très souvent, ils n'opèrent pas de manière indépendante mais s'associent aux autres. Certaines formes de fraude impliquent des structures complexes mises en place pour commettre des fraudes de manière organisée. Cela fait que la fraude n'est pas un événement isolé.

Ces caractéristiques constituent un défi très important pour les industries, principalement le secteur financier, pour détecter les activités frauduleuses telles que les transactions frauduleuses effectuées au moyen d'une carte de crédit ou d'un chèque. Il existe une liste exhaustive des fraudes décrites dans de nombreux ouvrages tels que [90] et [71].

2.6.3 Système de détection de fraude du monde réel

Les Systèmes de Détection de Fraude (FDS) peuvent être compliqués en raison des contraintes de la vie réelle. Dans cette section, nous décrivons les principales particularités et les conditions de fonctionnement d'un FDS réel, en nous basant sur la Figure 2.8 [30].

la Figure 2.8 décrit la hiérarchie des couches dans un FDS, chacune contrôlant si les transactions sont authentiques ou devraient plutôt être signalées comme une fraude : i) Terminal, ii) Règles de blocage des transactions, iii) Règles de notation, iv) Modèle basé

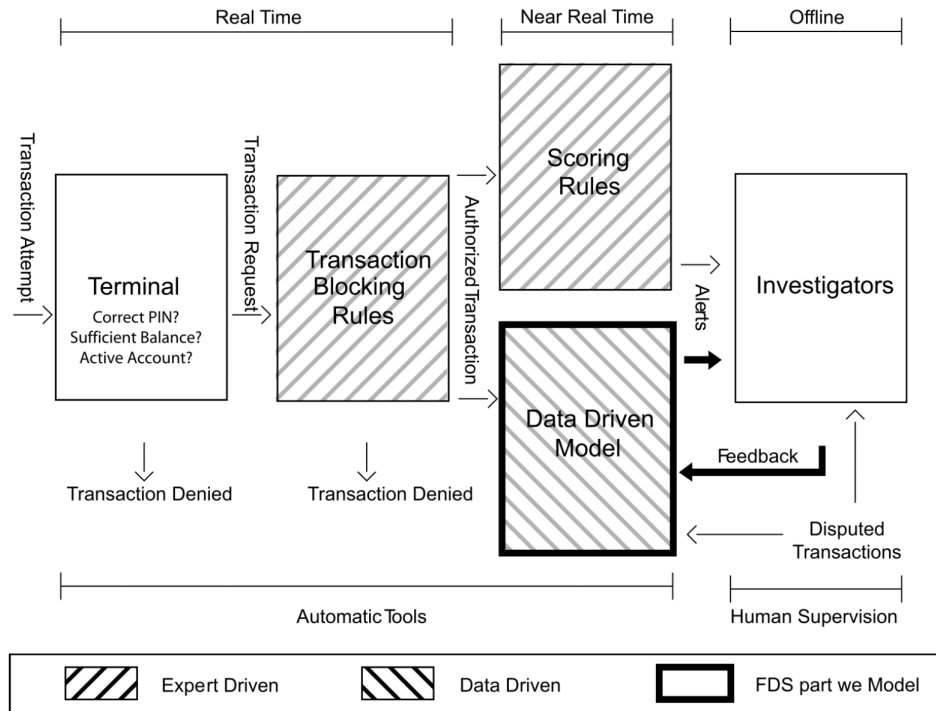


FIGURE 2.8 – Les couches d'un FDS([30]).

sur les données, v) Les enquêteurs. Les quatre premiers éléments du FDS sont entièrement automatisés, tandis que le dernier nécessite une intervention humaine et c'est la seule partie non automatique et hors ligne du FDS.

- (i) **Terminal** : Le terminal représente la première couche de contrôle dans un FDS et effectue des contrôles de sécurité classiques sur toutes les demandes de paiement. Les contrôles de sécurité incluent le contrôle du code PIN, possible uniquement dans le cas de cartes équipées de puce, le nombre de tentatives, l'état de la carte (active ou bloquée), le solde disponible et la limite de dépenses. En cas de transactions en ligne, ces opérations doivent être effectuées en temps réel et la réponse doit être fournie en quelques millisecondes, au cours desquelles le terminal interroge un serveur de la société émettrice de la carte. Les demandes qui ne passent aucun de ces contrôles sont refusées, tandis que les autres deviennent des requêtes de transaction qui sont traitées par la deuxième couche de contrôle.
- (ii) **Règles de blocage des transactions** : les règles de blocage des transactions sont des instructions **if-then (- else)** destinées à bloquer les demandes de transaction qui sont clairement perçues comme des fraudes. Ces règles utilisent les quelques informations disponibles lorsque le paiement est demandé, sans analyser les historiques ni le profil du titulaire de la carte. Un exemple peut être le suivant : « **If** une transaction basée sur un code PIN est effectuée en Europe **And** en Asie dans les 5 minutes à partir du même titulaire de carte **Then** refuser la transaction ». En effet, le client ne peut pas être physiquement en Europe et en Asie sur un court intervalle de temps. Notez qu'il

ne s'agit que d'un exemple illustratif, les règles de blocage sont des informations confidentielles. Les règles de blocage des transactions sont conçues manuellement par l'enquêteur et, en tant que telles, sont des composants du FDS pilotés par des experts.

- (iii) Les règles de notation sont également des modèles pilotés par des experts (EDR) définies par les enquêteurs en fonction de leur expérience et agissent comme un classifieur basé sur des règles. Ils attribuent un score à chaque transaction autorisée : plus le score est élevé, plus la transaction est susceptible d'être une fraude. Dans la pratique, les règles de notation contiennent des conditions simples et compréhensibles par l'homme. Un exemple de règle de notation peut être : « **If** transaction précédente dans un autre continent **And** moins de une heure après la transaction précédente **Then** score de fraude = 0,95) ». Les règles de notation ne peuvent détecter que les stratégies frauduleuses qui ont déjà été découvertes par les enquêteurs et qui présentent des modèles impliquant peu de composants des vecteurs de caractéristiques. De plus, les règles de notation sont plutôt subjectives, car différents experts conçoivent des règles différentes.
- (iv) Modèle basé sur les données (DDM) : cette couche est purement basée sur les données et s'appuie sur un modèle prédictif pour attribuer un score de fraude à chaque transaction. Habituellement, cette phase utilise des algorithmes d'apprentissage qui renvoient pour chaque transaction une estimation de la probabilité d'être une fraude. Ces algorithmes peuvent apprendre des corrélations complexes dans les données en utilisant un grand volume de données avec une dimensionnalité élevée. Ils sont généralement plus robustes que l'EDR, mais la plupart sont des boîtes noires, c'est-à-dire qu'il n'est pas possible de les convertir en règles faciles à interpréter. Un DDM est capable de prendre en compte toutes les informations associées à une transaction, tandis qu'un EDR renvoie des conditions basées sur quelques caractéristiques des transactions.
- (v) Les enquêteurs sont des experts en fraude qui vérifient les alertes de fraude, c'est-à-dire les transactions qui ont reçu un score de fraude élevé soit par EDR, soit par DDM. Ces transactions suspectes sont affichées dans un outil de gestion des cas (CMT) où les enquêteurs peuvent voir le score de fraude associé, vérifier leur origine (EDR ou DDM) et les annoter comme authentiques ou frauduleuses après vérification. Dans un scénario réaliste, seules quelques alertes peuvent être vérifiées compte tenu d'un nombre limité d'enquêteurs [30]. Pour cette raison, ils enquêtent généralement uniquement sur les transactions avec le score de fraude le plus élevé. Le rôle des enquêteurs est de contacter le titulaire de la carte, de vérifier si la transaction est frauduleuse ou non et de l'annoter avec son étiquette correcte. Ce processus peut être long, fastidieux et en même temps, le nombre de transactions à vérifier est généralement très important. Il est donc important de minimiser les fausses alertes. Le processus d'étiquetage génère des transactions annotées qui sont utilisées par le DDM.

2.7 Conclusion

Dans ce premier chapitre, nous avons présenté brièvement les différents domaines et concepts utilisés dans ce manuscrit de thèse. Nous avons introduit le domaine de la science des données et le problème de détection d'anomalies. Nous avons discuté de ce que sont les anomalies et pourquoi la détection des anomalies peut être très importante pour le traitement des données. Ensuite, nous avons fourni des connaissances préliminaires sur l'apprentissage automatique. Enfin, nous avons présenté le problème de détection de fraude qui est le domaine d'application de notre travail, accompagné d'une brève description du système réel de détection de fraude.

Dans le chapitre suivant, nous allons faire un état de l'art des différentes techniques d'apprentissage automatiques pour la détection de fraudes. Ensuite, nous décrivons le problème d'apprentissage à partir d'ensembles de données déséquilibrées. Enfin, nous décrivons les différentes approches proposées pour traiter le problème de déséquilibre.

Chapitre 3

État de l’art

Sommaire

3.1	Introduction	37
3.2	Mesures de performance des modèles d’apprentissage automatique	38
3.3	Approches basées sur l’apprentissage automatique pour la détec- tion des fraudes	42
3.3.1	Approches supervisées	42
3.3.2	Approches non supervisées	46
3.4	Apprentissage en distribution déséquilibrée	51
3.4.1	Distribution déséquilibrée : méthodes au niveau des données	52
3.4.2	Distribution déséquilibrée : méthodes au niveau de l’algorithme	56
3.5	Conclusion	61

3.1 Introduction

La détection de la fraude est un sujet très difficile et crucial qui est au centre de nombreuses recherches et études. L'analyse de la fraude est un processus consistant en une séquence de fonctions pour prédire ou découvrir des menaces potentielles ou explicites d'activités frauduleuses. Le processus repose sur des techniques provenant d'une grande variété de domaines, notamment l'exploration de données, les statistiques, l'apprentissage automatique, etc. L'efficacité d'un FDS dépend en grande partie de l'efficacité des techniques utilisées et des données pertinentes.

L'apprentissage automatique est considéré comme l'une des techniques les plus efficaces pour l'identification des fraudes. Il utilise une approche de classification et de régression pour reconnaître la fraude par carte de crédit. Le problème de détection de fraudes peut être divisé en deux catégories principales qui dépendent de la disponibilité ou non des étiquettes. Dans le premier cas, l'apprentissage automatique supervisé utilise des transactions étiquetées pour former le classifieur, tandis que dans le second cas, l'algorithme d'apprentissage non supervisé utilise une analyse de groupe de pairs qui regroupe les clients en fonction de leur profil et identifie la fraude en fonction du comportement de dépenses des clients.

Cependant, le nombre de transactions authentiques est bien plus élevé que le nombre de fraudes. Comme le jeu de données est très déséquilibré lors de l'application de l'apprentissage automatique directement aux données, la prédiction s'incline vers la classe majoritaire [93]. Ceci affecte fortement les performances de la détection. Par conséquent, trouver une bonne solution pour classer les transactions frauduleuses et les transactions normales avec une bonne précision est devenu un domaine de recherche important.

En matière de détection de fraudes, et pour résoudre ce problème de déséquilibre de données, les approches développées au cours des dernières années peuvent être classées comme suit. La première approche est une approche basée sur les données pour les méthodes d'échantillonnage. L'objectif de cette approche est de gérer ou de produire l'ensemble de données bien équilibré à partir de l'ensemble de données d'apprentissage déséquilibré. La deuxième approche est basée sur un algorithme. L'objectif de cette dernière est de modifier le classifieur pour accorder plus d'attention à la classe d'intérêt.

Ce chapitre se concentre sur l'état de l'art des approches qui ont été adoptées pour la détection de la fraude par carte de crédit. Il est structuré en trois sections. La première est dédiée à la présentation des métriques de performance pertinentes utilisées dans ce contexte. Dans la deuxième section, nous allons faire un tour d'horizon sur les différentes techniques d'apprentissage automatique pour la détection de fraudes. Deux grandes catégories d'algorithmes sont discutées. Dans la troisième section, nous présentons le problème d'apprentissage à partir d'ensembles de données déséquilibrées. Nous décrivons les méthodes proposées afin de remédier à ce problème. Ces différentes méthodes peuvent être regroupées en deux catégories : les méthodes opérantes au niveau des données, et

les méthodes opérantes au niveau algorithmique. Nous terminons ce chapitre par une conclusion.

3.2 Mesures de performance des modèles d'apprentissage automatique

Éléments fondamentaux pour mesurer la performance

Dans le domaine de l'apprentissage automatique, un modèle d'apprentissage apprend et prédit des labels dits positifs ou négatifs. La classe avec très peu d'instances d'apprentissage, mais une grande importance d'identification, est appelée la classe positive ; l'autre est appelée la classe négative. Les résultats attendus et les résultats obtenus par un modèle ML (supervisé ou non supervisé) sont alors organisés dans une matrice de confusion (voir le Tableau 3.1) composées des métriques principales suivantes :

TABLE 3.1 – Matrice de confusion

	Predicted as Positive	Predicted as Negative
Actually Positive	True Positives (TP)	False Negatives (FN)
Actually Negative	False Positive (FP)	True Negatives (TN)

- Vrai négatif ou True Negative (TN) : le nombre d'instances négatives correctement classés.
- Vrai positif ou True Positive (TP) : le nombre d'instances positives correctement classés.
- Faux positif ou False Positive (FP) : le nombre d'instances positives mal classés.
- Faux négatif ou False Negative (FN) : le nombre d'instances négatives mal classés.

Les mesures d'évaluation jouent un rôle crucial à la fois dans l'évaluation des performances des différents modèles ML et dans l'orientation de leur modélisation [108]. Trois particularités ont guidé le choix des mesures de performance pour l'évaluation des systèmes de détection de fraude :

- les limites de ressources d'un processus d'investigation manuel obligatoire : étant donné que l'équipe d'enquête sur les fraudes ne peut examiner que quelques centaines de cas suspects chaque jour ,
- le déséquilibre des classes : lorsque les instances négatives dépassent largement les instances positives, une mesure de performance appropriée doit tenir compte de ce préalable non uniforme ,

- l'asymétrie entre False Positive et False Negative : classer une instance positive comme négative peut entraîner des coûts plus élevés que de classer une instance négative comme positive. Dans les faits, l'inverse pourrait également être vrai, car les entreprises ont généralement des régimes d'assurance pour couvrir leurs rejets de débit alors qu'ils doivent maintenir une équipe d'enquêteurs ,

Comment mesurer la performance ?

L'exactitude est la mesure de performance la plus couramment utilisée pour évaluer un algorithme. Elle est définie comme un rapport entre les instances correctement classées et le nombre total d'instances. Elle est également appelé le complémentaire du taux d'erreur. En utilisant les notations du Tableau 3.1, l'exactitude peut être mesurée comme suit :

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (3.1)$$

$$Error_{rate} = \frac{FP + FN}{TP + FP + TN + FN} \quad (3.2)$$

Toutefois, face au problème de déséquilibre où une classe positive est représentée par seulement 1% des données d'apprentissage, un modèle simple peut prédire l'étiquette de classe négative pour chaque instance. Il peut atteindre une exactitude élevée de 99%. À partir de cet exemple, nous voyons que l'exactitude peut être sans importance dans le cadre de déséquilibre de classes. Plusieurs travaux de recherche s'intéressent à la mesure de performance des modèles d'apprentissage dans le cas de déséquilibre de données [10, 86, 110].

La spécificité, le taux vrai négatif (TNR) où le rappel inverse est exprimé comme le rapport des échantillons négatifs correctement classés au nombre total d'échantillons négatifs comme dans l'équation 3.3 [110].

$$TNR = \frac{TN}{FP + TN} \quad (3.3)$$

Le taux de faux négatifs (FNR) ou le taux d'omission est la proportion d'instances positives qui ont été mal classées. Ainsi, il complète la mesure de sensibilité et il est défini dans l'équation 3.4 [110].

$$FNR = \frac{FN}{FN + TP} \quad (3.4)$$

Le taux de faux positifs (FPR) est également appelé taux de fausses alarmes (FAR). Il représente le rapport entre les instances négatives mal classées et le nombre total d'instances négatives [110]. En d'autres termes, c'est la proportion des instances négatives qui ont été mal classées. Par conséquent, il complète la spécificité comme dans l'équation 3.4.

$$FPR = \frac{FP}{FP + TN} \quad (3.5)$$

La sensibilité, le taux vrai positif (TPR), où rappel représente les instances positives correctement classées par rapport au nombre total d'instances positives. Elle est estimée en fonction de l'équation 3.6 [110], avec $Rappel \in [0, 1]$ où $Rappel = 1$ est la meilleure valeur lorsque toutes les instances de la classe positive sont bien classées.

$$Rappel = \frac{TP}{TP + FN} \quad (3.6)$$

La précision ou la valeur de prédiction positive (PPV) représente la proportion d'instances positives correctement classées par rapport au nombre total d'instances positives prédites comme indiqué dans l'équation 3.7 [110], avec une $Precision \in [0, 1]$ et où la valeur $Precision = 1$ est atteinte lorsqu'il n'y a pas d'instances faux positives.

$$Precision = \frac{TP}{TP + FP} \quad (3.7)$$

Les deux métriques 3.6 3.7 partagent une relation inverse entre elles. Cependant, contrairement à l'exactitude et à l'erreur, la précision et le rappel ne sont pas sensibles aux changements dans la distribution des données. Un modèle parfait capturera toutes les instances positives ($Rappel = 1$) et notera comme seules les instances qui sont en fait ($Precision = 1$), d'un point de vue analytique, il est souhaitable d'augmenter le rappel sans sacrifier la précision.

F-mesure ou F1-score représente la moyenne harmonique de précision et de rappel comme dans l'équation 3.8 [110]. La valeur de la mesure F est comprise entre zéro et un, et des valeurs élevées de la mesure F indiquent des performances de prédiction élevées.

$$F1\text{-score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3.8)$$

La courbe des caractéristiques de fonctionnement du récepteur (ROC) est un graphique bidimensionnel dans lequel le TPR représente l'axe y et le FPR est l'axe x à divers réglages du seuil. La Figure 3.1 montre un exemple de la courbe ROC [110] comme suit :

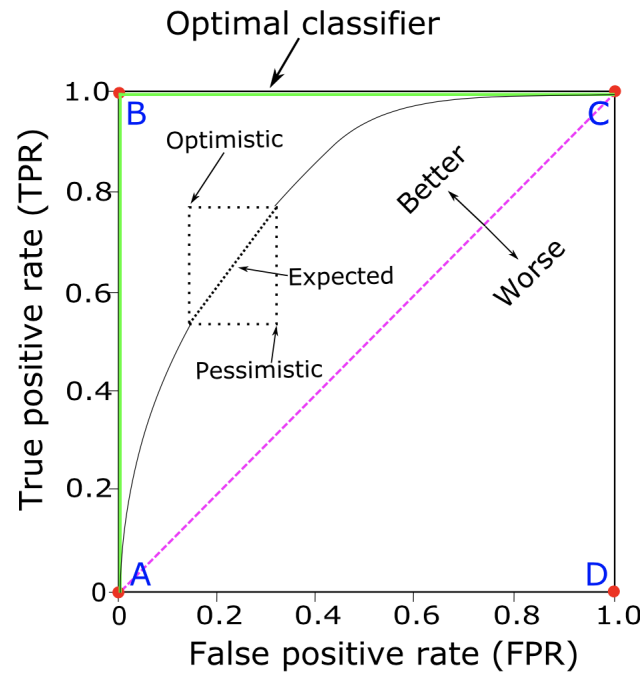


FIGURE 3.1 – Une courbe ROC de base montrant les points importants [110]

- le point A, dans le coin inférieur gauche (0,0) représente un classifieur où il n'y a pas de classification positive, alors que tous les instances négatives sont correctement classées et donc $TPR = 0$ et $FPR = 0$;
- le point B dans le coin supérieur gauche (0,1) représente un classifieur où tous les instances positives et négatives sont correctement classées. Ainsi, ce point représente la classification parfaite ou le point de fonctionnement idéal ;
- le point C, dans le coin supérieur droit (1,1), représente un classifieur où tous les instances positives sont correctement classées, tandis que les instances négatives sont mal classées ;
- le point D dans le coin inférieur droit (1,0) représente un classifieur où tous les instances positives et négatives sont mal classées.

La Figure 3.1 montre les performances de classification parfaites. C'est la courbe verte qui monte verticalement de (0,0) à (0,1) puis horizontalement à (1,1). Cette courbe montre que le classifieur a parfaitement classé les instances positives par rapport aux instances négatives. Un point dans l'espace ROC est meilleur que tous les autres points situés au sud-est, c'est-à-dire les points qui ont un TPR inférieur, un FPR plus élevé ou les deux (voir Figure 3.1). Par conséquent, tout classifieur apparaît dans le triangle inférieur droit fonctionne moins bien que le classifieur qui apparaît dans le triangle supérieur gauche.

L'aire sous la courbe ROC (Area under curve, AUC) est un indicateur synthétique de la courbe ROC (Figure 3.1). Il est utilisé pour calculer l'aire sous la courbe ROC et estimé en fonction de l'équation 3.9. Le score AUC est toujours limité entre zéro et un.

Plus la valeur de l'AUC est proche de 1, plus le modèle est meilleur.

$$AUC = \frac{1 + TPR - FPR}{2} \quad (3.9)$$

La communauté de recherche a largement abandonné les mesures simples telles que l'exactitude, le taux de classification erronée ou le taux réel positif. Elle a plutôt adopté l'aire sous la courbe caractéristique de l'opérateur du récepteur (AUC) pour résumer la performance d'un système sur une plage de seuils de décision sous classes déséquilibrées [90, 67, 93].

3.3 Approches basées sur l'apprentissage automatique pour la détection des fraudes

Les activités frauduleuses entraînent des pertes importantes, ce qui a motivé la recherche dans ce domaine à trouver une solution permettant de détecter et de prévenir les fraudes. Plusieurs méthodes basées sur l'apprentissage automatique (ML) ont été proposées et testées. Ces méthodes de détection de la fraude ont été divisées en deux grandes catégories : supervisée et non supervisée. Dans les méthodes de détection de fraude supervisée, les modèles sont estimés sur la base des instances de transactions légitimes et frauduleuses pour classer les nouvelles transactions comme légitimes ou frauduleuses. Tandis que dans la détection de fraude non supervisée, les transactions aberrantes sont détectées comme des cas potentiels de transactions frauduleuses. Dans ce qui suit, nous présentons quelques importantes approches supervisées et non supervisées.

3.3.1 Approches supervisées

Les Approches supervisées reposent sur l'ensemble des transactions passées pour lesquelles l'étiquette (également appelée classe) de la transaction est connue. Dans les problèmes de détection de fraude par carte de crédit, l'étiquette est soit légitime (la transaction a été effectuée par le titulaire de la carte), soit frauduleuse (la transaction a été effectuée par un fraudeur). L'étiquette est généralement connue a posteriori, soit parce qu'un client s'est plaint, soit à la suite d'une enquête. Les techniques supervisées utilisent des transactions passées étiquetées pour apprendre un modèle de prédiction de fraude, qui renvoie, pour toute nouvelle transaction, la probabilité qu'il s'agisse d'une fraude [32, 18]. Les méthodes supervisées ont dominé la littérature sur la détection des fraudes.

Dans [102], les auteurs utilisent des modèles supervisés basés sur les méthodes d'arbre de décision et un certain nombre de méthodes SVM différentes avec des noyaux de fonctions de base polynomiale, sigmoïde, linéaire et radiale a été réalisée. L'objectif de ce travail est de démontrer les avantages de l'application des techniques d'apprentissage automatique, y compris les arbres de décision et les SVM, au problème de détection de

fraude par carte de crédit dans le but de réduire le risque. Les données de carte de crédit utilisées dans cette étude sont extraites des entrepôts de données de carte de crédit d'une banque disposant des autorisations requises. La distribution des données est fortement déséquilibrée avec un rapport d'environ 20000 enregistrements de transactions normales pour un enregistrement de transaction frauduleuse. Les méthodes utilisées sont appliquées sur trois échantillons avec des ratios d'enregistrement frauduleux/normaux différents. Le premier ensemble d'échantillons a un rapport d'un enregistrement frauduleux à un enregistrement normal; le second a un rapport d'un enregistrement frauduleux à quatre enregistrements normaux; et le dernier a le rapport d'un frauduleux à neuf normaux. Les auteurs ont comparé les performances de ces méthodes en utilisant l'exactitude des résultats de performance. L'étude a montré que les méthodes de DT surpassent les méthodes de SVM.

Une autre étude [101], utilisant d'autres modèles supervisés avec des paramètres d'entrée différents sur un ensemble de données réelles de carte de crédit d'une banque, basés sur les réseaux de neurones artificiels (ANN) et la régression logistique (LR). La distribution des données est fortement déséquilibrée avec un rapport d'environ 20000 enregistrements de transaction normale pour un enregistrement de transaction frauduleuse. Trois ensembles différents sont formés pour comparer les performances des méthodes dans ces différentes conditions. Dans le premier ensemble, il y a une transaction normale pour chaque transaction frauduleuse. Dans le deuxième ensemble, il y a quatre transactions normales pour chaque frauduleuse et il y en a neuf normales pour chaque frauduleuse dans le troisième ensemble. L'exactitude et le taux du vrai positif (TPR) qui montre le ratio de transactions frauduleuses capturées sont utilisés comme mesures pour évaluer les performances des modèles. Les résultats montrent que les modèles ANN surpassent les modèles LR pour résoudre le problème étudié.

Dans [106], une approche basée sur SVM (support vector machine) avec implication de plusieurs noyaux de fonctions de base (linéaire, quadratique et radiale) a été appliquée pour la détection de fraude par carte de crédit. Ils ont généré les données de transactions normales et frauduleuses en utilisant différentes moyennes et variances, puis les ont mélangées avec des probabilités différentes. Leurs résultats montrent que le noyau de la fonction de base radiale surpasse le noyau linéaire et quadratique dans tous les domaines de comparaison. Ainsi, il a eu une exactitude maximale jusqu'à 97%, TPR maximum (99%), TNR maximum (98%) et FPR maximum (7%), FNR maximum (6%). Il se comporte également presque de la même manière pour tous les types des ensembles de données générés, ayant des données de fraude très faibles et des données de fraude élevées.

Une comparaison expérimentale de plusieurs algorithmes et techniques de modélisation de pointe a été réalisée sur un ensemble de données de fraude par carte de crédit réel fourni par un prestataire de services de paiement en Belgique [32]. Le but de cette étude était de donner quelques lignes directrices aux praticiens sur la façon d'aborder le problème de détection de fraude, en se concentrant sur trois questions cruciales : le déséquilibre, la non-stationnarité et l'évaluation. Les auteurs ont testé trois algorithmes supervisés différents : Random Forest (RF), Neural Network (NNET) et Support Vector Machine (SVM). Ils ont proposé la précision moyenne (AP), l'aire sous la courbe ROC (AUC) et PrecisionRank comme mesures de performance correctes pour une tâche de détection de

fraude. En matière de résultats, ils ont conclu que Random Forests surpasse clairement ses concurrents.

APATE (abréviation de : Anomaly Prevention using Advanced Transaction Exploration) a été proposé par [115]. Il s'agit d'une approche automatisée et temps réel pour lutter contre la fraude aux transactions par carte de crédit en cartographiant les modèles d'achat passés et le comportement des clients en fonctionnalités significatives et en comparant ces fonctionnalités caractéristiques à toute nouvelle transaction entrante. Les auteurs ont appliqué des techniques d'apprentissages automatiques supervisés (Logistic Regression (LR), Random Forest (RF), et Neural Network (NNET)) pour découvrir des modèles frauduleux à partir d'un ensemble de données de transaction par carte de crédit réel obtenu auprès d'un grand émetteur de cartes de crédit en Belgique. Les résultats montrent que l'approche proposée conduit à un score de l'aire sous la courbe ROC (AUC) et une exactitude très élevés, en particulier pour le modèle de forêt aléatoire.

Dans [9], une approche hybride de détection de fraude par carte de crédit dans laquelle la détection de fraude se fait en trois phases a été proposée. La première phase concerne l'authentification initiale de l'utilisateur et la vérification des détails de la carte. Si le chèque est effacé avec succès, alors la transaction est passée à la phase suivante où l'algorithme de clustering Fuzzy C-means est appliqué pour découvrir les types de transactions normaux des utilisateurs de cartes de crédit en fonction de leur activité passée. Un score de suspicion est calculé en fonction de l'étendue de l'écart par rapport aux types normaux et, par conséquent, la transaction est classée comme légitime, suspecte ou frauduleuse. Une fois qu'une transaction est jugée suspecte, un mécanisme d'apprentissage basé sur un réseau neuronal est appliqué pour déterminer s'il s'agit d'une activité frauduleuse ou d'une déviation occasionnelle par un utilisateur légitime. En raison de l'indisponibilité des ensembles de données de carte de crédit réels ou de tout ensemble de données de référence, ce travail a utilisé la distribution gaussienne pour générer les transactions synthétiques qui représentent le comportement des véritables utilisateurs ainsi que des fraudeurs. L'ensemble des données est divisé en trois catégories soit 70% pour la phase de formation, 15% pour la validation, 15% pour les tests. La couche d'apprentissage classe les transactions suspectes et donne les résultats suivants : 93,90% de transactions correctement classées et 6,10% de transactions incorrectement classées.

Un cadre de détection de fraude basé sur Convolutional Neural Networks (CNNET) a été proposé dans [53]. Les auteurs souhaitent capturer les modèles intrinsèques de comportements de fraude appris à partir de données étiquetées. Le cadre comprend des parties de formation et de prédiction. La partie formation comprend principalement quatre modules : l'ingénierie des caractéristiques à l'aide des stratégies d'agrégation, la méthode d'échantillonnage basée sur les coûts pour générer des fraudes synthétiques, la transformation des caractéristiques en une matrice de caractéristiques pour former le modèle et une procédure de formation basée sur CNN. La partie formation est hors ligne et la partie prédiction est en ligne. Lorsqu'une transaction intervient, la partie prédiction peut juger immédiatement si elle est frauduleuse ou légitime. Pour évaluer le cadre proposé, un ensemble de données de transaction par carte de crédit réel provenant d'une banque commerciale a été utilisé. Il contient plus de 260 millions de transactions de cartes de crédit en un an, environ 4000 transactions sont qualifiées fraudes et les autres sont des transactions légitimes. Le score

F1 est utilisé comme mesure de performance. D'après les résultats, le modèle CNN sur différents ensembles d'échantillons atteint les meilleures performances en comparant avec d'autres modèles existants.

Dans [19], on aborde le problème de la détection des fraudes et on offre un aperçu du processus complet de développement et de mise en œuvre d'un système de détection de fraude chez un grand commerçant de commerce électronique. Ce travail compare différentes méthodes d'apprentissage automatique : Logistic Regression (LR), Support Vector Machine (SVM) et Random Forest (RF) en utilisant deux métriques l'aire sous la courbe de ROC (AUC) et l'aire sous la courbe de précision-recall (AUPR) sur un ensemble de données réel de l'étude de cas de l'entreprise. À partir des résultats, les auteurs ont confirmé que les méthodes d'apprentissage supervisé sont applicables à la détection de fraude dans un environnement de commerce électronique. Les trois algorithmes d'apprentissage automatique testés (LR, SVM et RF) ont donné de bons résultats. RF a obtenu les performances les plus élevées parmi les trois. Ils ont conclu que RF semble être très appropriée pour être utilisée pour la détection de fraude, non seulement en raison de ses bonnes performances, mais aussi en raison de la facilité de mise en œuvre et du temps de calcul rapide, même avec de grands ensembles de données.

Dans [70], les auteurs présentent une approche de détection de fraude utilisant une méthode Naive Bayes (NB) et K-Nearest Neighbours (KNN). Le principal objectif du système proposé est d'améliorer l'exactitude de détection. Le classifieur NB prédit les probabilités de fraude dans les transactions tandis que le classifieur KNN prédit à quel point les données d'échantillon non définies sont proches du même ensemble de données d'entraînement. Les auteurs ont comparé ces classifieurs NB et KNN et ont montré que les deux fonctionnent différemment pour un ensemble de données de cartes de crédit de Kaggle avec une exactitude d'environ 95% et 90% respectivement. Ils ont conclu que la détection des fraudes de carte de crédit en utilisant un seul algorithme individuel ne sera pas une pratique efficace. Comme chaque algorithme contient ses propres avantages et inconvénients et afin de détecter efficacement la fraude par carte de crédit, il faut combiner des algorithmes afin de pouvoir tirer parti de chaque algorithme.

Une approche de fusion a été proposée [103] comprenant le classifieur Naive Bayes (NB) et des algorithmes de clustering (Expectation Maximization (EM), Kmeans) en utilisant les données simulées pour détecter la fraude dans les transactions par carte de crédit. Les données simulées ont été composées à l'aide de trois composants : les données sur les étiquettes de fraude et les heures d'arrivée des transactions ont été simulées par le processus de Markov Modulated Poisson, les montants des transactions pour les transactions authentiques ont été simulés à l'aide de la distribution gaussienne et les montants des transactions pour les Distribution gaussienne. Les algorithmes de clustering Expectation Maximization (EM) avec trois initialisations différentes et K-means ont été appliqués pour regrouper les montants de transaction en trois groupes (élevé, moyen et faible), puis à deux groupes (élevé et faible). L'algorithme NB a ensuite été appliqué pour classer les transactions comme légitimes ou frauduleuses. Pour évaluer les performances du FDS, trois mesures communes ont été prises. Le taux de vrais positifs (TPR), les taux de faux positifs (FPR) et l'exactitude. D'après les résultats, parmi les trois mesures de performance, l'algorithme de clustering EM est meilleur que l'algorithme K-means. Il est recommandé

d'augmenter le nombre de groupes, ce qui peut aider à déterminer la meilleure initialisation de l'algorithme de regroupement EM et à dresser le profil des dépenses des titulaires de cartes de crédit et des fraudeurs.

Dans [122], les auteurs examinent les performances de deux types de modèles Random Forest (RF) qui sont différents dans leurs classificateurs de base. Un classifieur de base de Random Forest I, qui est une implémentation simple de l'arbre de décision, appelé un arbre aléatoire. Le classifieur de base de Random Forest II est CART. es de données, il existe encore des problèmes tels que des données déséquilibrées. La principale différence entre les deux algorithmes est la manière de fractionnement des noeuds. L'ensemble des données utilisé dans cette expérience provient d'une société de commerce électronique de Chine et se compose de transactions frauduleuses et légitimes de novembre 2016 à janvier 2017. Il contient plus de 30 millions de transactions individuelles pour 82 miles transactions qui ont été qualifiées fraudes (taux de fraude de 0,27%). D'après les résultats obtenus, les auteurs ont conclu que, bien que Random Forest obtient de bons résultats, il a encore des problèmes tels que des données déséquilibrées.

3.3.2 Approches non supervisées

Les approches d'apprentissages supervisées reposent sur l'hypothèse que les modèles légitimes et frauduleux tirés des données passées se répercutent dans le futur. Cependant, lorsque les comportements des clients ou des fraudeurs changent, les schémas appris et les relations entre eux peuvent ne plus permettre de prendre des décisions précises. Pour cette raison, les techniques non supervisées ne nécessitent pas de connaissances préalables des transactions légitimes et frauduleuses pour construire un modèle. Elles visent à caractériser la distribution des données des transactions. Elles reposent sur l'hypothèse que les valeurs aberrantes de la distribution des transactions sont des fraudes, elles peuvent donc être utilisées pour détecter des types de fraudes invisibles car elles ne reposent pas sur des transactions qualifiées de frauduleuses dans le passé [18].

L'un des principaux avantages de l'utilisation des techniques non supervisées dans le système de détection de fraude par carte est la possibilité de trouver des modèles frauduleux non découverts. Cependant, ces approches sont moins courantes. Dans cette partie, on présente différentes techniques proposées utilisant des approches d'apprentissages non supervisées.

Un système automatisé de détection de fraude par carte de crédit peut aussi être basé sur une approche de réseaux neuronaux, comme le montre les auteurs dans [124]. Ces auteurs ont appliqué le Self-Organizing Map (SOM), un réseau neuronal non supervisé, pour créer un modèle de comportement typique du titulaire de carte et pour analyser la déviation des transactions, trouvant ainsi des transactions suspectes. Les données transactionnelles sont des informations confidentielles, par conséquent, l'ensemble de données initial a été simulé (une liste de paramètres de transactions réelles et une plage de leurs valeurs ont été utilisées). Comme résultats, ils ont conclu que le succès de l'algorithme ne dépend pas d'hypothèses statistiques sur la distribution des données. Il traite avec succès les données bruyantes. Le procédé permet de modifier le modèle au fur et à mesure que de

nouvelles transactions sont ajoutées et il ne nécessite pas d'informations a priori en plus d'un ensemble de transactions effectuées par le titulaire de la carte. La précision obtenue des règles produites est stable avec la possibilité de détecter des structures de données isolées.

Dans [36], trois méthodes sont utilisées pour détecter les fraudes. La première utilise le modèle de clustering pour classer la transaction légale et frauduleuse à l'aide de la classification des données des régions. La deuxième utilise la Gaussian Mixture Model (GMM) pour modéliser la densité de probabilité du comportement passé de l'utilisateur de la carte de crédit afin que la probabilité du comportement actuel puisse être calculée pour détecter toute anomalie du comportement passé. Enfin, la troisième se base sur Bayesian Network (BN) pour décrire les statistiques d'un utilisateur particulier et les statistiques de différents scénarios de fraudes. L'un des objectifs de cette étude est d'identifier le modèle d'utilisateur qui identifie le mieux les cas de fraudes. Les modèles sont comparés en termes de leurs performances. En conclusion, pour améliorer le système de détection des fraudes, les auteurs suggèrent la combinaison des trois méthodes présentées.

Un autre travail [87] présente une méthodologie de détection efficace pour identifier les utilisateurs légitimes et retracer leurs activités illégales à l'aide de Hidden Markov Model (HMM). Ce travail se base sur l'algorithme baum-welch et montre comment il peut être utilisé pour la détection des fraudes. Le modèle HMM est initialement formé avec le comportement normal d'un titulaire de compte. Si une transaction bancaire en ligne entrante n'est pas acceptée par le HMM formé avec une probabilité suffisamment élevée, elle est considérée comme frauduleuse. Dans le même temps, ils veillent à ce que les transactions authentiques ne soient pas rejetées. Les résultats expérimentaux révèlent que l'exactitude du système est proche de 72% sur une large variation des données d'entrée et démontrent l'utilité du modèle de Markov caché. Le système est également évolutif pour traiter de gros volumes de transactions.

Les transactions par carte de crédit étant des données temporelles, l'adaptabilité des outils de détection de fraudes à l'évolution du comportement (concept drift) des fraudeurs doit être abordée afin de garantir l'efficacité continue d'une approche de détection de fraudes. Dans [16], on présente un cadre non supervisé, les étiquettes de classes sont inconnues, qui nécessite des adaptations à la présence de dérives conceptuelles pour l'analyse des flux de données. Les auteurs ont proposé une méthode de représentation synthétique basée sur l'apprentissage d'une SOM pondérée (Self-Organizing Map) et une mesure de dissimilarité entre ces représentations pour la détection de changement dans la structure du flux, afin de détecter différents types de fraudes dans une période de temps. L'approche proposée a été validée sur une application réelle de détection de fraudes par carte de crédit en ligne. Les données utilisées pour ce test contiennent 76 332 transactions frauduleuses décrites par 64 variables. Ils ont montré à travers quelques exemples que la méthode est capable de détecter des changements dans la structure du flux et de détecter des dérives de concept.

Dans [114], les auteurs utilisent une approche de clustering pour la détection de la fraude par carte de crédit. Comme l'ensemble de données réelles n'est pas disponible, les données sont générées de manière aléatoire (ID de transaction, montant, pays, date,

numéro de carte de crédit, catégorie de commerçant, type de transaction, ID de cluster, *isfraud*, *isnewtransaction*). L'algorithme de clustering K-Means est appliqué sur ces données. Pour classer une transaction comme frauduleuse ou légitime, quatre clusters sont formés : faible, élevé, risqué et à haut risque. À partir des résultats expérimentaux, les auteurs ont constaté que l'algorithme de clustering une fois mis en œuvre montre des résultats significatifs. La plupart des activités frauduleuses ont pu être correctement identifiées. Cependant, il y a eu un certain nombre d'activités non frauduleuses, qui ont été identifiées à tort comme des fraudes. Pour détecter la fraude avec précision et efficacité, il est nécessaire que les données réelles soient disponibles.

Dans [8], les auteurs ont comparé deux approches basées sur les algorithmes non supervisés Self-Organizing Map (SOM) et Induction Decision Tree (ID3). Les approches ont regroupé les données en quatre groupes : faible, élevé, risqué et très risqué. Les deux méthodes ont été testées dans quatre ensembles de données comprenant 500, 100, 1500 et 2000 transactions, mais aucune information supplémentaire n'a été spécifiée sur les données. SOM avait un FPR légèrement meilleur (23,52% et 28% respectivement) et 20% meilleur TPR que ID3 (92,5% et 72,5%). En outre, les auteurs concluent qu'en utilisant l'ensemble des données le plus grand, le FPR s'est amélioré de 50% et le TPR de 20%.

Un système anti-fraude est développé en utilisant une combinaison de deux algorithmes non supervisés : l'analyse en composantes principales et l'algorithme K-means [81]. Pour pouvoir tester l'efficacité du système proposé, des données réelles sont évidemment nécessaires. Néanmoins, la collecte de ces données réelles auprès de différentes banques échoue généralement, car elle est souvent liée à des transactions financières sensibles gardées confidentielles pour des raisons élémentaires de confidentialité. Des données aléatoires et falsifiées sont donc générées à cette fin. Selon différents tests, le modèle proposé classe les transactions avec une bonne précision et il peut détecter de nouveaux comportements frauduleux.

Dans le monde financier dont fait partie le réseau des crypto-monnaies, la détection d'anomalies équivaut à une détection de fraude. Dans [88], on étudie l'utilisation de k-means pour détecter les activités frauduleuses dans les transactions Bitcoin. Cette étude a utilisé l'ensemble de données Bitcoin hébergé par le Laboratoire de biologie computationnelle de l'Université de l'Illinois. Toutes les transactions du bloc genesis à la blockchain 230686 du 7 avril 2013. Les objets contenus par le jeu de données du réseau Bitcoin ne sont pas étiquetés, c'est pourquoi cette étude opte pour des algorithmes capables de détecter des valeurs aberrantes en considérant la structure sous-jacente des regroupements existants au sein du réseau. Les résultats prouvent que l'algorithme a détecté avec succès certaines des activités frauduleuses connues.

Dans [2], on propose une nouvelle approche de la surveillance des transactions et de la détection de la fraude par carte de crédit en utilisant un apprentissage non supervisé. Les auteurs utilisent la SOM pondérée (Self-Organizing Map), pour détecter les transactions frauduleuses à l'aide d'enregistrements du montant et des détails de localisation des transactions précédentes effectuées par les clients. Une fois les clusters obtenus à l'aide de SOM, les auteurs revalident les clusters en utilisant des règles d'association sur chaque cluster.

Dans [123], on propose une méthode de détection de fraude non supervisée utilisant le clustering basé sur l'auto-encodeur. Un auto-encodeur avec trois couches cachées et un clustering k-means est utilisée et testée sur un ensemble de données de transactions de banques européennes de Kaggle. Dans cette recherche, le but est de détecter les transactions frauduleuses de manière non supervisée pour améliorer la précision de la détection des fraudes à une vitesse logique. Sur la base des résultats, la précision de cette méthode était de 98,9%, ainsi qu'un TPR de 81%.

Dans [97], les auteurs présentent un système de détection de fraudes basé sur un réseau neuronal (NN). Ce travail porte sur la détection de fraudes dans les données de carte de crédit par une approche apprentissage automatique non supervisée. Les performances sont mesurées sur diverses mesures : précision, rappel, taux de faux positifs, spécificité, exactitude équilibrée, score F1 et exactitude. Pour identifier les valeurs aberrantes, les auteurs ont évalué les performances avec d'autres algorithmes sur l'ensemble de données de carte de crédit de Kaggle : Auto Encoder (AE), Local Outlier Factor (LOF), Isolation Forest (IF) and K-means clustering. Les résultats expérimentaux montrent que l'approche proposée basée sur un réseau de neurones offre une précision de 99,98%, tandis que les précisions de AE, IF, LOF et de K-means sont respectivement de 97%, 98%, 98% et 99,97%.

Le Tableau 3.2 synthétise les techniques des deux catégories d'apprentissage automatique pour la détection de fraude examinées dans cette section. En outre, les avantages et les inconvénients de chaque catégorie des techniques examinées sont résumés.

TABLE 3.2 – Résumé des algorithmes d'apprentissage automatique pour la détection des fraudes

Apprentissage Automatique	Méthodes	Références	Avantages	Inconvénients
Apprentissage supervisé	Neural Network	[101, 32, 115, 9, 53]	Très précis et fiable	- Besoin de comprendre et d'étiqueter l'entrée - Plus de temps de calcul requis pour la phase d'apprentissage
	Support Vector Machine	[102, 106, 32, 19]		
	K-Nearest Neighbor	[70]		
	Logistic Regression	[101, 115, 19]		
	Decision Tree	[102]		
	Random Forest	[32, 115, 19, 122]		
	Bayesian Network	[70, 103]		
Apprentissage non supervisé	K-Means	[103, 97, 114, 88, 81]	- Facile à trouver des modèles et des caractéristiques inconnus des données	- Calcul complexe - Moins précis en raison d'une entrée non étiquetée
	Isolation Forest (IF)	[97]		
	Local Outlier Factor (LOF)	[97]		
	Fuzzy C-Means	[9]		
	Hidden Markov Model (HMM)	[87]		
	Self-Organizing Map (SOM)	[124, 16, 8, 2]		
	Auto Encoder (AE)	[97]		
	Gaussian Mixture Model (GMM)	[36]		

En résumé, de nombreuses techniques de ML ont été utilisées pour traiter la fraude par carte de crédit. La plupart des techniques proposées ont donné des résultats prometteurs.

D'après cette étude, nous concluons que l'apprentissage supervisé est largement utilisé pour le système de détection de fraude plutôt que pour le non supervisé. La raison intuitive est que dans le domaine des cartes de crédit, il est facile d'annoter l'étiquette d'une transaction une fois qu'un titulaire de carte a déposé une plainte. Nous avons observé que les techniques de détection de fraude supervisées les plus couramment utilisées sont Neural Network (NNET), Random Forest (RF), Support Vector Machine (SVM). Les techniques non supervisées les plus couramment utilisées sont K-means, et Self-Organizing Map(SOM).

Cependant, la plupart des ensembles de données utilisés sont très déséquilibrés. La plupart des ensembles de données contenaient un pourcentage plus élevé de transactions légales par rapport aux transactions frauduleuses. Ce déséquilibre des données a fait de la détection de la fraude une tâche difficile à résoudre. En outre, la plupart des techniques proposées n'ont pas été testées sur un ensemble de données du monde réel ; ils ont été testés sur un ensemble de données généré artificiellement. En effet, la plupart des institutions financières ne publient pas l'ensemble de données en raison d'un accord de confidentialité qu'elles ont signé avec leurs clients. En revanche, les systèmes de détection de fraude par carte de crédit traitent généralement des millions de transactions. Par conséquent, pour améliorer les performances de classification des systèmes de détection de fraude, il existe un besoin d'un environnement pour gérer efficacement la grande quantité et la variété des données.

Pour traiter le problème mentionné dont souffrent les techniques de l'apprentissage automatique, nous réaliserons une évaluation expérimentale de performances des techniques d'apprentissage automatique supervisées et non supervisées pour la détection de fraude, afin d'identifier leurs faiblesses lorsqu'ils sont confrontés au problème de déséquilibre de classes dans un environnement Big Data [41].

3.4 Apprentissage en distribution déséquilibrée

Avec l'intérêt croissant pour l'adoption de l'apprentissage automatique (ML) à des problèmes du monde réel, les effets des données déséquilibrées sont devenus évidents. Traditionnellement, la plupart des algorithmes ML supposent que l'ensemble de données utilisé pour apprendre soit équilibré. Cependant, s'agissant des applications réelles, cet équilibre n'est pas garanti.

Les données sont dites déséquilibrées lorsqu'au moins une classe est sous représentée par rapport aux autres [85, 31, 41]. La classe d'intérêt est généralement la classe minoritaire, car dans la réalité, les événements rares sont souvent ceux qui ont le plus d'intérêt. Le problème de données déséquilibrées est considéré comme l'un des problèmes les plus critiques rencontrés par les systèmes de détection de fraudes et d'anomalies. L'apprentissage d'un algorithme ML, dans un tel contexte, a un impact sérieux sur les performances qui ont tendance à être dépassés par la classe majoritaire et à ignorer la classe minoritaire [23, 83, 31, 41].

Un certain nombre d'approches ML dans un contexte de données déséquilibrées ont vu le jour. On distingue alors deux niveaux de traitement : au niveau des données et au niveau des algorithmes [23]. Au niveau des données, ces travaux incluent de nombreuses formes différentes de ré-échantillonnage comme le sur-échantillonnage, et le sous-échantillonnage pour rééquilibrer les données avant l'application de tout algorithme [82, 96, 125, 38]. Au niveau algorithmes, ces derniers sont ajustés afin de contrer le déséquilibre de classes [107, 17, 20, 54].

3.4.1 Distribution déséquilibrée : méthodes au niveau des données

Les méthodes au niveau des données se concentrent sur le prétraitement des données avant l'entraînement pour corriger le déséquilibre, afin que les algorithmes ML puissent augmenter leurs performances. Ces méthodes utilisent des approches d'échantillonnage pour produire un ensemble de données bien équilibré à partir de l'ensemble de données déséquilibré. Deux approches sont utilisées pour créer un ensemble de données équilibré qui sont le sous-échantillonnage [96, 82, 117] et le sur-échantillonnage [63, 15, 38, 72].

Méthodes au niveau des données : sous-échantillonnage

L'une des stratégies les plus courantes et les plus simples pour équilibrer un ensemble de données consiste à éliminer au hasard des échantillons de la classe majoritaire jusqu'à ce qu'à ce que les classes minoritaire et majoritaire soient équilibrées. Cette méthode est appelée sous-échantillonnage aléatoire [61]. Cependant, cela peut conduire à la suppression des instances informatives de la classe majoritaire, en particulier si le nombre d'instances de la classe minoritaire est très faible. Ainsi, il y aura une perte importante des instances de la classe majoritaire, conduisant à une classification non optimale. D'autres stratégies appelées sous-échantillonnage dirigé sont développées pour contrôler les données supprimées de l'ensemble de données de départ.

Une première méthode dirigée est introduite par Hart en 1968 basée sur l'algorithme Nearest Neighbor. Elle est appelée Condensed Nearest Neighbor (CNN) [56]. CNN fonctionne en éliminant les échantillons de la classe majoritaire éloignés de la frontière de décision, car ces échantillons peuvent être considérés comme moins pertinents pour l'apprentissage. Quelques années plus tard, Ivan Tomek a proposé deux modifications de l'algorithme Condensed Nearest Neighbor (CNN), dont l'une est appelée lien Tomek (TL) [113]. Cette approche fonctionne en détectant une paire de points de données, appelés lien de Tomek, qui sont les plus proches voisins mais ont des étiquettes de classe différentes. Le sous-échantillonnage peut être effectué en supprimant tous les liens Tomek ou en supprimant les données de la classe majoritaire appartenant au lien Tomek.

Une autre méthode de sous-échantillonnage appelée Edited Nearest Neighbor (ENN) est introduite dans [121]. Elle suit le même principe que CNN et vise à supprimer toute instance dont l'étiquette de classe est différente de la classe d'au moins deux de ses trois voisins les plus proches. L'idée derrière cette technique est de supprimer les instances de

la classe majoritaire à proximité ou autour de la limite des différentes classes, afin d'augmenter la précision de classification des instances minoritaires plutôt que des instances majoritaires. Un autre travail similaire, appelée Neighborhood Cleaning Rule (NCR) a été proposée dans [79]. Il utilise la règle Edited Nearest Neighbor proposée dans [121] pour supprimer des instances de la classe majoritaire lorsque deux des trois voisins les plus proches d'une instance contredisent la classe.

Deux améliorations de ENN sont introduites dans [112] : Repeated Edited Nearest Neighbor (RENN) et All-KNN (AKNN). Les deux méthodes effectuent plusieurs passes sur l'ensemble d'entraînement en répétant ENN. RENN répète simplement l'algorithme ENN jusqu'à ce que aucune autre élimination ne puisse être effectuée à partir de l'ensemble édité. AKNN répète ENN pour chaque échantillon en utilisant des valeurs d'incrémentations de k à chaque fois et en supprimant l'échantillon si son étiquette n'est pas la prédominante au moins pour une valeur de k .

Une autre approche de sous-échantillonnage, appelée One-Side Selection (OSS) [77], résulte de la combinaison de l'application successive de Tomek Link (TL) [113] et de l'algorithme Condensed Nearest Neighbor (CNN) [56]. Le lien de Tomek est utilisé pour supprimer les exemples de la classe majoritaire bruyante et borderline. Ensuite, CNN supprime les exemples de la classe majoritaire éloignés de la frontière de décision. Ensuite, un sous-ensemble cohérent de la classe majoritaire est formé.

Dans [96], les auteurs proposent une technique de sous-échantillonnage basée sur le clustering en considérant le problème du déséquilibre de classe pour les données cardiovasculaires. L'objectif est de réduire l'écart entre les échantillons de la classe majoritaire et les échantillons de la classe minoritaire. Les auteurs ont regroupé les échantillons de la classe majoritaire en k -clusters. Après cela, chaque cluster est combiné séparément avec les échantillons de la classe minoritaire pour créer k différents ensembles de données d'apprentissage. Enfin, tous les ensembles de données combinés sont classés avec un algorithme d'apprentissage et l'ensemble de données avec la plus grande précision est utilisé pour construire le modèle d'apprentissage.

Deux stratégies de sous-échantillonnage sont introduites [82]. Elles sont basées sur une technique de clustering pour sous-échantillonner la classe majoritaire afin de produire le même nombre d'échantillons de données que la classe minoritaire. Dans la première stratégie, les centres de cluster sont utilisés pour remplacer l'ensemble des données de la classe majoritaire. Alors que pour la deuxième stratégie, on utilise les voisins les plus proches de chaque centre de cluster.

Dans [117], on introduit un cadre de sous-échantillonnage pour gérer le déséquilibre de classe dans les ensembles de données binaires en supprimant les points de données potentiellement superposés. On conçoit quatre méthodes basées sur la recherche de voisinage avec différents critères pour identifier et éliminer les instances de classe majoritaire de la région de chevauchement.

Dans la Figure 3.2, nous illustrons sur un ensemble de données le comportement de certaines stratégies de sous-échantillonnage. Nous notons que la procédure CNN a supprimé un grand nombre d'exemples de la classe majoritaire. D'autre part, en raison de la rareté de la classe minoritaire dans l'ensemble de données, nous n'avons pas été en mesure de

supprimer un grand nombre d'exemples de la classe majoritaire à l'aide de la procédure Tomek Link.

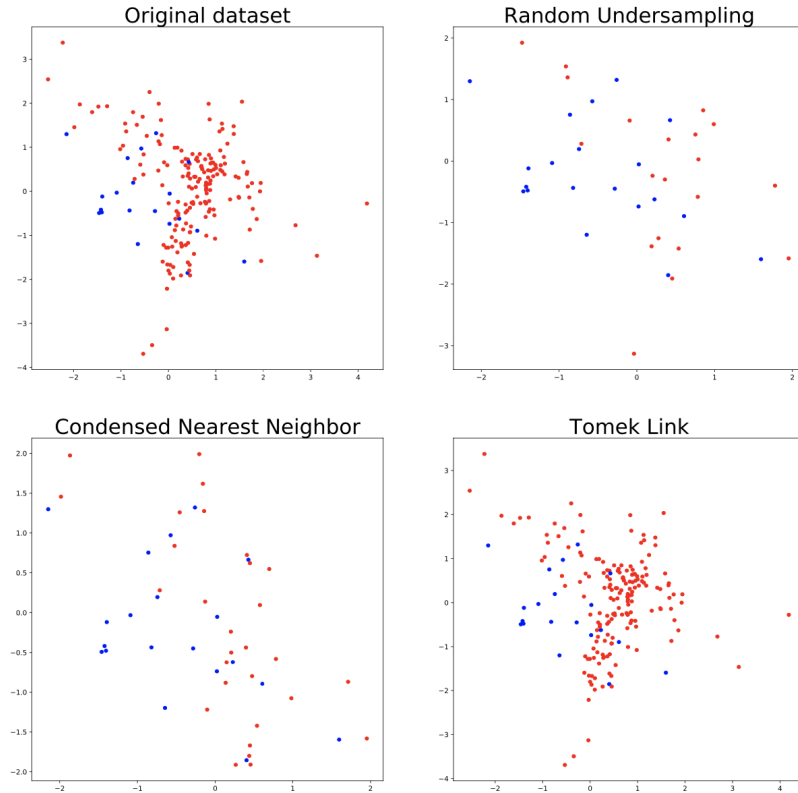


FIGURE 3.2 – Illustration de certaines stratégies de sous-échantillonnage

Méthodes au niveau des données : sur-échantillonnage

Contrairement au sous-échantillonnage, le sur-échantillonnage consiste à augmenter le nombre d'instances de la classe minoritaire. Cela peut se faire de différentes façons. L'approche courante est le sur-échantillonnage aléatoire (ROS), où les instances de la classe minoritaire sont reproduites pour garantir un équilibre entre la classe minoritaire et majoritaire. Le problème est qu'il existe une forte probabilité de sur-apprentissage ou de sur-ajustement en raison du fait que les mêmes instances se produisent plusieurs fois.

Pour éviter ce problème de sur-apprentissage, une nouvelle méthode de sur-échantillonnage principalement basée sur kNN [22] est proposée. Elle est appelée Synthetic Minority Oversampling Technique (SMOTE). Dans SMOTE, la classe minoritaire est sur-échantillonnée en prenant chaque échantillon de la classe minoritaire et en introduisant des exemples synthétiques le long des segments de ligne joignant tous les k voisins les plus proches de la classe minoritaire. En fonction de la quantité de sur-échantillonnage requise, les voisins des k voisins les plus proches sont choisis au hasard.

Dans [68], les auteurs présentent une approche appelée Clustering based oversampling (CBOS). Cette approche utilise un sur-échantillonnage basé sur le clustering. Ils utilisent la

technique de clustering K-means. Avant le suréchantillonnage, les échantillons des classes minoritaires et majoritaires doivent être regroupés. Dans la classe majoritaire, tous les clusters, à l'exception du plus grand, sont suréchantillonnés au hasard afin d'obtenir un nombre comparable d'échantillons d'apprentissage en tant que plus grand cluster. Dans la classe minoritaire, toutes les grappes sont suréchantillonnées au hasard pour obtenir le même nombre jusqu'à ce que la taille de l'échantillon de la minorité soit égale à celle de la majorité.

Dans [60], on propose une approche de sur-échantillonnage, dite ADAPtive SYNthetic (ADASYN), pour apprendre à partir d'ensembles de données déséquilibrés. ADASYN est similaire à SMOTE, comme le montre la Figure 3.3. La différence réside dans l'utilisation d'une distribution de densité comme critère de décision automatique du nombre d'échantillons synthétiques qui doivent être générés pour chaque exemple minoritaire, et ce en changeant de manière adaptative les poids des différents exemples minoritaires pour compenser les distributions asymétriques.

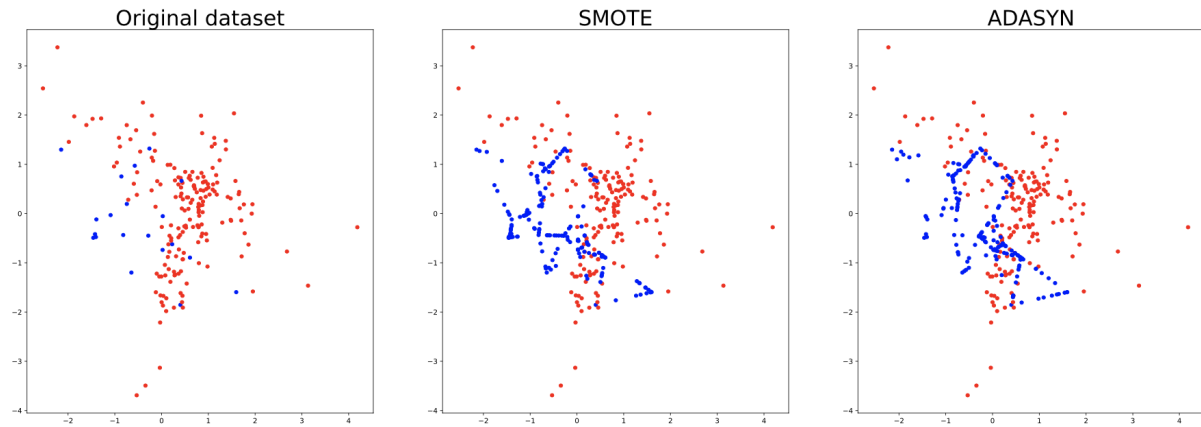


FIGURE 3.3 – Illustration de certaines stratégies de sur-échantillonnage

Dans [63] on trouve une variante de l'algorithme SMOTE appelée Modified Synthetic Minority Over-sampling Technique (MSMOTE). Cette approche prend également en compte la distribution des instances de la classe minoritaire. Il divise les données appartenant à la classe minoritaire en trois catégories : border, noise et latent selon leur éloignement de toutes les instances d'apprentissage.

Une autre méthode appelée Safe-level SMOTE (SSMOTE) est introduite dans [15]. Elle est basée sur SMOTE. l'objectif principal de SSMOTE est d'éviter la génération d'échantillons bruyants. Chaque échantillon minoritaire reçoit d'abord une valeur de niveau de sécurité, puis seuls les échantillons avec des valeurs de niveau de sécurité plus élevées sont sélectionnés pour générer de nouveaux échantillons synthétiques minoritaires. La valeur du niveau de sécurité de chaque échantillon minoritaire est définie comme le nombre d'autres échantillons minoritaires parmi ses k voisins les plus proches.

Dans [38], on étudie une méthode de sur-échantillonnage basée sur le clustering k-means et SMOTE. Cette méthode évite la génération de bruit et surmonte les déséquilibres

entre et au sein des classes. Les résultats d'expériences avec 90 jeux de données montrent que les données d'entraînement sur-échantillonnées avec la méthode proposée améliorent les résultats de la classification. Récemment, une méthode de sur-échantillonnage radial (RBO) a été proposée [72], elle consiste à trouver des régions dans lesquelles des objets synthétiques de la classe minoritaire devraient être générés sur la base de l'estimation de la distribution du déséquilibre avec les fonctions « radial basic ».

3.4.2 Distribution déséquilibrée : méthodes au niveau de l'algorithme

Les méthodes orientées algorithmes se concentrent sur la modification des algorithmes existants pour traiter le problème de déséquilibre de classe sans changer la distribution de classe de l'ensemble d'apprentissage. Deux approches principales peuvent être distinguées :

1. apprentissage d'ensemble [69, 62, 54, 12] : tente d'améliorer l'exactitude de la classe minoritaire ;
2. apprentissage sensible aux coûts [107, 17, 20, 75] : se concentre sur la minimisation des coûts de classification.

Méthodes au niveau de l'algorithme : apprentissage d'ensemble

L'apprentissage d'ensemble implique l'amélioration des performances des algorithmes ML uniques en induisant plusieurs algorithmes qui peuvent être identiques ou différents et en les combinant pour créer un nouveau algorithme. Par conséquent, l'idée fondamentale est de créer plusieurs algorithmes à partir des données de départ, puis d'agréger leurs prédictions lorsque des instances inconnues sont présentées. Les méthodes d'ensemble les plus connues sont le Boosting [69, 24, 105, 54] et le Bagging [13, 62, 12].

Le Boosting est un algorithme itératif qui consiste à attribuer différents poids aux instances dans l'ensemble de données d'apprentissage. Après chaque itération, le poids sur les instances mal classées augmente et sur les instances correctement classées diminue. Les erreurs étant souvent concentrées sur les classes rares, on pourrait penser que le Boosting améliore l'apprentissage sur des ensembles de données déséquilibrés en augmentant le poids des instances appartenant à la classe minoritaire. Différentes variantes de Boosting ont été proposées pour résoudre les problèmes de déséquilibre des données.

L'algorithme appelé AdaBoost de [49] est l'algorithme le plus représentatif. Il s'agit du premier algorithme de Boosting pratique, et reste l'un des plus largement utilisés et étudiés, avec des applications dans de nombreux domaines.

Dans [69], les auteurs ont proposé un algorithme de boosting appelé : RareBoost. Cet algorithme applique la règle suivante : si le nombre de vrais positifs est supérieur au nombre de faux positifs, le poids des instances bien classées diminue. Le poids des instances mal classées augmente si le nombre de vrais négatifs est supérieur à celui de faux négatifs.

SMOTEBoost, proposé dans [24], combine l'algorithme SMOTE avec AdaBoost, résultant en un algorithme hybride d'échantillonnage et de Boosting qui surpasse SMOTE

et AdaBoost. Une alternative plus simple et plus rapide à SMOTEBoost est proposée dans [105]. Ce travail combine le Boosting et le sous-échantillonnage de données appelé RUSBoost. Il est conçu pour améliorer les performances des modèles entraînés sur des données biaisées.

Une autre approche d'ensemble variante de RUSBoost [105] combine le sous-échantillonnage aléatoire avec un algorithme de Boosting. Aussi, l'approche appelée EUSBoost [54] vise à améliorer les propositions existantes en améliorant les performances des classifieurs de base par l'utilisation de l'approche de sous-échantillonnage évolutif. Cette approche favorise la diversité par l'utilisation de différents sous-ensembles d'instances de classe majoritaire pour former chaque classifieur de base.

Le Bagging, également appelé agrégation bootstrap, est un méta-algorithme d'ensemble d'apprentissage automatique. Il est basé sur un processus stochastique de mise à jour de l'ensemble de données d'entraînement pour créer un ensemble diversifié de classifieurs. Cette méthode consiste à construire chaque classifieur à l'aide d'un échantillon d'instances issues du jeu de données d'origine avec un remplacement. Afin de garantir un nombre suffisant d'instances par classifieur, tout en considérant chaque instance avec un poids égal, chaque échantillon contient généralement le même nombre d'instances que dans l'ensemble de données d'origine. Enfin, lorsqu'une instance inconnue est présentée à chaque classifieur individuel, un vote majoritaire ou pondéré est utilisé pour déduire la classe.

Une méthode globale d'apprentissage automatique connue sous le nom de Random Forest (RF) est proposée par [13]. L'auteur utilise plusieurs arbres comme classifieurs en utilisant le Bagging. Après avoir obtenu le vote majoritaire sur tous les classifieurs, la méthode RF combine les informations sur tous les arbres pour révéler les variables importantes.

Dans [62], les auteurs introduisent un algorithme d'ensemble appelé : Bagging Roughly Balanced (RB). Il utilise une nouvelle technique d'échantillonnage pour améliorer l'algorithme de Bagging original pour les ensembles de données avec des distributions de classes déséquilibrées. Pour cette méthode d'échantillonnage, le nombre d'échantillons dans les classes les plus grandes et les plus petites est différent, mais ils sont effectivement équilibrés lors de la moyenne de tous les sous-ensembles.

Un autre type de Bagging appelé Neighborhood Balanced Bagging (NBBag) est proposé [12]. Il est basé sur différentes étapes. Au lieu d'intégrer le Bagging au prétraitement, l'idée de Bagging standard est conservée mais parfois radicalement. Les probabilités d'échantillonnage des exemples en Bagging sont modifiées en augmentant les chances de dessiner des exemples minoritaires. De plus, le rôle des exemples minoritaires difficiles par rapport au type de leur quartier est amplifié. La force d'amplification peut être paramétrée dans leur réglage.

Méthodes au niveau de l'algorithme : apprentissage sensible aux coûts

Ces algorithmes sont sensibles à différents coûts associés à certaines caractéristiques des problèmes considérés. Les coûts peuvent provenir de divers aspects liés à un problème

réel donné et peuvent être fournis par un expert du domaine ou appris pendant la phase de formation du classifieur.

De nombreuses méthodes sensibles au coût ont été proposées. Un cadre d'apprentissage général sensible aux coûts, appelé : MetaCost a été introduit dans [37]. Il consiste à apprendre un modèle en appliquant une procédure de minimisation des coûts, qui utilise un algorithme d'apprentissage de base. Ensuite, MetaCost estime les probabilités de classe à l'aide du bagging, puis ré-étiquette les instances d'entraînement avec leurs classes de coût minimales attendues. Enfin, il entraîne un modèle à l'aide de l'ensemble d'apprentissage modifié.

Une autre approche sensible aux coûts est appelée : Adacost [46]. Dans cette approche, les exemples appartenant à la classe rare qui sont mal classés se voient attribuer des poids plus élevés que ceux appartenant à la classe commune. Il est démontré empiriquement que le système proposé produit des coûts cumulatifs d'erreurs de classification inférieurs.

Dans [127], les auteurs étudient l'effet de l'échantillonnage et du déplacement de seuil dans la formation de réseaux de neurones sensibles aux coûts. Une technique de déplacement de seuil a été utilisée pour déplacer le seuil de sortie vers des classes peu coûteuses, de sorte que les instances avec des coûts plus élevés deviennent plus difficiles à mal classer.

Dans [107], des algorithmes de Boosting sensibles aux coûts pour améliorer la classification des données déséquilibrées sont examinés. Les auteurs ont proposé trois algorithmes de Boosting sensibles aux coûts appelés AdaC1, AdaC2 et AdaC3 en introduisant des éléments de coût dans le cadre de l'apprentissage dans la formule de mise à jour du poids d'AdaBoost.

Le framework proposé dans [17] intègre directement la mesure de l'évaluation Area Under the ROC Curve (AUC) et G-mean dans la fonction objective de SVM (Support Vector Machine) sensible au coût. L'objectif principal est d'améliorer les performances de la classification en optimisant simultanément la meilleure paire de sous-ensembles de fonctionnalités et les paramètres de coût d'une classification erronée.

Les auteurs de [20] ont étudié le problème du déséquilibre de classe dans le contexte des réseaux de neurones en utilisant MultiLayer Perceptron (MLP). Un algorithme sensible aux coûts appelé (CSMLP) est proposé pour améliorer la capacité discriminatoire MLP (deux classes). La formulation du CSMLP est basée sur une fonction objective commune qui utilise un seul paramètre de coût pour distinguer l'importance des erreurs de classe.

Les auteurs de [75] ont introduit un ensemble d'arbres de décision sensibles aux coûts pour une classification déséquilibrée. Il est basé sur la combinaison d'arbres de décision sensibles aux coûts avec un partitionnement aléatoire de l'espace des fonctionnalités basé sur un sous-espace. L'objectif des auteurs est d'assurer la création d'un pool de classifieurs individuels capables d'améliorer la reconnaissance de la classe minoritaire.

Le Tableau 3.3 résume les approches proposées dans la littérature pour traiter le problème du déséquilibre de classe et leurs avantages et inconvénients.

TABLE 3.3 – Résumé des approches de traitement de données déséquilibrées

Techniques		Références	Avantages	Inconvénients
Méthodes au niveau des données	Sous-échantillonnage	[61, 56, 113, 121, 79, 112, 77, 96, 82, 117]	- Technique simple et largement utilisée dans de nombreux domaines d'applications - Elle peut être facilement implémentée - Elle peut aider à améliorer le problème associé au temps d'exécution et au stockage, simplement en éliminant les instances des données de formation	- Elle entraîne la perte d'informations utiles et significatifs - Elle peut entraîner des performances médiocres sur des données invisibles
	Sur-échantillonnage	[22, 68, 60, 63, 15, 38, 72]	- Elle peut être facilement mis en œuvre - Elle n'entraîne aucune perte d'information	- Elle augmente la probabilité de sur-apprentissage - Elle introduit un coût de calcul supplémentaire et prend du temps de formation

TABLE 3.3 – Résumé des approches de traitement de données déséquilibrées

Techniques		Références	Avantages	Inconvénients
Méthodes au niveau de l'algorithme	Apprentissage d'ensemble	[49, 69, 24, 105, 105, 54, 13, 62, 12]	- Plus de résilience au bruit - Elle évite le problème de sur-ajustement - Elle donne de meilleures performances de prédictions par rapport à un seul classifieur	- Ajouter plus de classifieurs augmente la complexité - Elle prend du temps de formation
	Apprentissage sensible aux coûts	[37, 46, 127, 107, 17, 20, 75]	- Technique simple et efficace - Elle minimise le coût de la classification erronée (en biaisant le classifieur vers la classe minoritaire)	- Les coûts de classification erronée (le coût réel des erreurs) sont souvent inconnus

Les travaux discutés dans cette section couvrent un large éventail de solutions pour le déséquilibre de classe au niveau des données et au niveau de l'algorithme. Cependant, nous reconnaissons qu'il existe des lacunes importantes dans la recherche actuelle qui manque de méthodes de recherche, plus particulièrement pour la détection d'anomalies ou fraude, sur des sujets tels que le chevauchement des classes, les données bruitées, le déséquilibre dans un environnement de big data. Toutes les méthodes présentées discutent du problème de déséquilibre, c'est-à-dire le déséquilibre dans la fréquence de classe. Cependant, Un autre phénomène critique dans les données déséquilibrées dans le monde réel concerne relativement la présence d'un nombre d'instances de la classe majoritaire située à l'intérieur de la classe minoritaire. Ce problème est communément appelé chevauchement de classes et nous l'avons appelé « bruit comportemental ». Le problème du chevauchement entre les classes apparaît lorsqu'une région de l'espace de données contient une quantité similaire de données d'apprentissage de chaque classe. Ce problème peut devenir catastrophique pour l'apprentissage lorsqu'il est en conjonction avec le déséquilibre. Dans de nombreux cas il est même plus perturbant que le déséquilibre lui-même. Pour traiter les problèmes mentionnés dont souffrent les études analysées ci-dessus, nous allons proposer une nouvelle approche appelée : One Side Behavioral Noise Reduction (OSBNR) [44, 43].

3.5 Conclusion

L'intelligence artificielle et l'apprentissage automatique fournissent des modèles efficaces pour la détection de fraudes, qui ont été appliquées avec succès, et peuvent être largement classées en deux grandes catégories : supervisées et non supervisées. Par ailleurs, la fraude est un évènement rare en proportion des volumes de transactions légitimes, l'ordre de grandeur est habituellement d'une transaction frauduleuse sur mille. En conséquence l'ensemble des données d'apprentissage utilisées pour construire les modèles de détection est très fortement déséquilibré, qui peut conduire à une mauvaise classification des données.

Dans ce chapitre bibliographique, nous avons introduit, d'abord, plusieurs métriques de performance bien adaptées dans ce contexte. Ensuite, nous avons donné un aperçu sur les approches ML qui ont été adoptées pour la détection de fraude par carte de crédit. Nous avons discuté les deux grandes catégories avec leurs avantages et inconvénients. Puis, nous avons cité les différentes techniques proposées dans la littérature pour traiter le problème de déséquilibre, qui sont regroupées en deux grandes familles : les méthodes opérant au niveau de données et les méthodes opérant au niveau algorithmique.

Conclusion

Dans cette partie, nous avons abordé le contexte, les fondations nécessaires et certains travaux connexes.

Dans le chapitre. 2, Nous avons introduit le domaine de la science des données et le problème de détection d'anomalies. Nous avons discuté de ce que sont les anomalies et pourquoi la détection des anomalies peut être très importante pour le traitement des données. Ensuite, nous avons fourni des connaissances préliminaires sur l'apprentissage automatique. Enfin, nous avons présenté le problème de détection de fraude qui est le domaine d'application de notre travail, accompagné d'une brève description du système réel de détection de fraude.

Dans le chapitre 3, nous avons donné un aperçu sur les approches ML qui ont été adoptées pour la détection de fraude par carte de crédit. Nous avons discuté les deux grandes catégories avec leurs avantages et inconvénients. D'après notre étude, nous concluons que l'apprentissage supervisé est largement utilisé pour le système de détection de fraude plutôt que pour le non supervisé. De plus, la plupart des ensembles de données utilisés sont très déséquilibrés, ce qui fait de la détection de la fraude une tâche difficile à résoudre. En revanche, les systèmes de détection de fraude par carte de crédit traitent généralement des millions de transactions. Par conséquent, pour améliorer les performances de classification des systèmes de détection de fraude, il existe un besoin d'un environnement pour gérer efficacement la grande quantité et la variété des données. Puis, nous avons cité les différentes techniques proposées dans la littérature pour traiter le problème de déséquilibre, qui sont regroupées en deux grandes familles : les méthodes opérant au niveau de données et les méthodes opérant au niveau algorithmique. Toutes les méthodes discutées discutent du problème de déséquilibre, c'est-à-dire le déséquilibre dans la fréquence de classe. Cependant, Un autre phénomène critique dans les données déséquilibrées dans le monde réel concerne relativement la présence d'un nombre d'instances de la classe majoritaire située à l'intérieur de la classe minoritaire. Ce problème est communément appelé chevauchement de classes et nous l'avons appelé « bruit comportemental ».

Dans la partie suivante, nous proposons des contributions pour traiter les deux problèmes mentionnés dont souffrent les techniques de l'apprentissage automatique.

Partie II

Contributions

Table des matières

Chapitre 4

Étude de performance des techniques d'apprentissage automatique dans une distribution déséquilibrée des données.

4.1	Introduction	66
4.2	Apprentissage automatique pour la détection d'anomalies	67
4.3	Évaluation expérimentale de performances de détection	74
4.4	Conclusion	86

Chapitre 5

Une approche basée sur l'apprentissage automatique pour traiter le problème de déséquilibre de classes en présence de bruit comportemental.

5.1	Introduction	88
5.2	Le principe de notre approche : OSBNR	89
5.3	Étude expérimentale	97
5.4	Conclusion	110

Chapitre 6

Système de détection de fraudes basé sur l'apprentissage automatique. Une approche méthodologique pour un système en production.

6.1	Introduction	112
6.2	Verrous liées aux données dans le système de detection d'anomalie . . .	114
6.3	Méthodologie d'apprentissage automatique et application à la détection des fraudes	115
6.4	Credit card fraud detection use case : Experimentation and Evaluation	118
6.5	Plateforme de développement de modèles ML	121
6.6	Conclusion	125

Chapitre 4

Étude de performance des techniques d'apprentissage automatique dans une distribution déséquilibrée des données.

Sommaire

4.1	Introduction	66
4.2	Apprentissage automatique pour la détection d'anomalies	67
4.2.1	Contexte et problématique	67
4.2.2	Approche de détection d'anomalies supervisée	68
4.2.3	Approche de détection d'anomalies non supervisée	72
4.2.4	Approche générale	73
4.3	Évaluation expérimentale de performances de détection	74
4.3.1	Description de données	74
4.3.2	Protocole Expérimental	77
4.3.3	Analyse des Résultats	79
4.4	Conclusion	86

4.1 Introduction

La détection d'anomalies est souvent menée à bien grâce à des techniques d'apprentissage automatique. Plusieurs travaux illustrant ces techniques ont été décrits dans le chapitre précédent 3. Les algorithmes d'apprentissage automatique sont divisés en deux types, supervisé et non supervisé. L'algorithme d'apprentissage supervisé utilise des données étiquetées, tandis que l'algorithme d'apprentissage non supervisé utilise une analyse de regroupement qui regroupe les données en fonction de certains critères de similarité.

En règle générale, ces techniques ont été conçues pour des ensembles de données équilibrés ou qui reposent sur certaines hypothèses sur la distribution des données. Cependant, ils sont confrontés aux particularités des données contenues dans des ensembles de données réelles. Parmi les difficultés les plus connues, nous citons les problèmes résultant du grand nombre de cas traités, les contraintes liées aux grandes dimensions, la sensibilité des algorithmes aux données bruitées et enfin le traitement des données lorsque la composante des classes présentées est déséquilibrée, ce qui est souvent le cas pour les données réelles.

Comme mentionné au chapitre précédent³, la revue de la littérature peut ne pas être suffisante pour découvrir l'efficacité observable des techniques d'apprentissage automatique. Ainsi, nous avons mené une étude expérimentale de ces techniques. La principale contribution de ce chapitre est que nous avons évalué de manière approfondie les modèles d'apprentissage supervisé et non supervisé pour la détection d'anomalies, en particulier dans les transactions bancaires par carte de crédit et nous avons évalué leurs performances sur un ensemble de données de transactions par carte de crédit Kaggle⁴ en tenant compte de la distribution des anomalies dans un ensemble de données déséquilibré.

Ce chapitre est structuré en deux sections. Tout d'abord, nous fournissons une description des techniques d'apprentissage automatique utilisées dans nos travaux pour la détection d'anomalies. Dans la deuxième section, une évaluation expérimentale de performances de ces techniques est fournie. Dans cette évaluation, nous visons à identifier les faiblesses des techniques d'apprentissage automatique lorsqu'elles sont confrontées au problème de déséquilibre de classes. Notre objectif est de détecter les problèmes qui doivent être résolus pour produire une approche hautement efficace pour le problème de déséquilibre de classe.

4. Plateforme la plus populaire pour les compétitions de science des données

4.2 Apprentissage automatique pour la détection d'anomalies

4.2.1 Contexte et problématique

La qualité des données est un problème difficile dans de nombreux domaines tels que la médecine, l'e-commerce ou les infrastructures informatiques. L'évaluation de la qualité des données nécessite le déploiement d'un certain nombre de services de données fondamentaux, y compris la détection d'anomalies. Les anomalies peuvent se manifester de différentes manières - par exemple via des valeurs manquantes ou des valeurs aberrantes - et peuvent être causées par des procédures erronées, des pannes système ou des événements inattendus. En général, la détection d'anomalies consiste à détecter des événements rares ou des données significativement différentes de la normale. Il a été largement appliqué dans d'innombrables domaines d'application. Parmi les exemples d'applications courantes, dans les données de transaction bancaire, une anomalie signifie qu'il peut y avoir eu une transaction frauduleuse [35, 118]. Dans le domaine de la sécurité informatique, la détection des anomalies peut être utilisée pour surveiller le trafic réseau et identifier les intrusions [26]. Les anomalies dans les données médicales peuvent être surveillées pour fournir des avertissements sanitaires préventifs [45].

Au cours des dernières années, plusieurs approches de détection d'anomalies ont été développées, y compris des approches basées sur l'intelligence artificielle utilisant des algorithmes d'apprentissage automatique. Notre étude de la littérature (discutée au chapitre 2) montre que la détection d'anomalies en apprentissage automatique peut être réalisée selon deux paradigmes qui dépendent fortement de la disponibilité des étiquettes. Pour les événements où chaque observation est associée à une étiquette (normale ou anormale), la détection d'anomalie est dite supervisée. Dans ce mode, l'objectif est de former un modèle pour mieux séparer les données normales des données anormales. Le deuxième paradigme est non supervisé, dans lequel aucune information n'est disponible a priori. L'approche consiste à construire un modèle de normalité basé uniquement sur des hypothèses de proximité ou de densité. Par exemple, un événement est considéré comme anormal et est censé être différent des autres, s'il est éloigné de son voisinage le plus proche ou s'il se trouve dans des régions à faible densité dans l'espace de description [21].

L'applicabilité de ces approches dépend fortement de la distribution des données. Plus précisément, la plupart des systèmes de détection d'anomalies ont été traditionnellement conçus pour des ensembles de données équilibrés ou qui ont certaines hypothèses sur la distribution des données. Cependant, les applications réelles sont plus souvent confrontées à plusieurs défis. L'un des principaux défis de la détection des anomalies est le déséquilibre de classe, où les données normales sont présentes en grande quantité et les instances anormales sont généralement très peu nombreuses. Un ensemble de données est dit déséquilibré lorsqu'une classe est sous-représentée (classe minoritaire) par rapport à une autre (classe majoritaire). L'exemple classique du déséquilibre est encore une fois la fraude transactionnelle, qui est heureusement beaucoup moins nombreuse par rapport au nombre de transactions légitimes effectuées par des millions de clients chaque jour. Cela rend la dé-

tection des anomalies semblable à la recherche de l'aiguille dans une botte de foin. Le fait d'avoir peu d'instances d'une classe signifie que l'algorithme d'apprentissage sera souvent incapable de généraliser le comportement de la classe minoritaire, ce qui aura un impact sur ses performances finales [31]. Cependant, la précision de prédiction de la classe minoritaire est d'une importance cruciale car cette classe est généralement d'un grand intérêt comme dans le cas de fraude. Ainsi, une mauvaise prédiction de la classe minoritaire peut avoir un coût beaucoup plus élevé par rapport à une mauvaise prédiction d'une instance de classe majoritaire.

En raison de ce défi, le problème de détection d'anomalies, dans sa forme la plus générale, n'est pas facile à résoudre. Dans ce contexte, nous nous intéressons aux approches d'apprentissage automatique appliquées à la détection d'anomalies survenant dans les transactions de données financières. Dans notre étude, nous utilisons l'ensemble de données fourni par Kaggle⁵. La particularité du problème est que les anomalies constituant la classe minoritaire sont représentées par un très faible pourcentage de l'ensemble de données. Pour étudier ce problème, nous développons une configuration expérimentale basée sur des algorithmes d'apprentissage automatique dans le cadre de la bibliothèque Machine Learning d'Apache Spark. L'objectif principal de cette étude est d'étudier les performances des techniques d'apprentissage automatique supervisé et non supervisé pour la détection d'anomalies et identifier leurs faiblesses ou inconvénients, lorsqu'elles sont confrontées au problème du déséquilibre de classes. Dans le but de détecter les problèmes qui doivent être résolus pour produire une approche hautement efficace pour le problème du déséquilibre de classe.

4.2.2 Approche de détection d'anomalies supervisée

Les techniques de détection d'anomalies en mode supervisé supposent la disponibilité d'un ensemble de données d'apprentissage contenant des instances étiquetées pour les classes normale et anormale.

Dans cette section, nous présentons les méthodes d'apprentissage supervisé utilisées dans le domaine de la détection d'anomalies et considérées dans ce travail. Plus précisément, nous nous intéressons aux méthodes ayant une implémentation officielle dans le cadre de calcul distribué open source Apache Spark.

Logistic Regression (LR) : parfois appelée modèle logistique ou modèle logit, a été développée par le statisticien David Cox en 1958. LR analyse la relation entre plusieurs variables indépendantes prédictives x et une variable dépendante catégorielle, et estime la probabilité d'occurrence d'un événement en ajustant les données à une courbe logistique. La fonction logistique oblige les probabilités estimées à se situer entre 0 et 1. L'équation 4.1 représente la définition de la fonction logistique générale.

$$\log\left[\frac{p}{1-p}\right] = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \dots + \beta_p x_{i,p} \quad (4.1)$$

5. Lien vers l'ensemble de données Kaggle : <https://www.kaggle.com/mlg-ulb/creditcardfraud>

où p est la probabilité estimée du résultat d'intérêt, β_0 est appelé le terme d'interception, $\beta_1, \dots, \beta_i$ sont appelés les coefficients de régression associés aux prédicteurs $x_1, \dots, x_i$ en conséquence, et i indique le i ème prédicteur. Sur la base de la probabilité estimée, un seuil de classification peut être défini pour déterminer deux groupes cibles et tous les cas sont classés dans les groupes de catégories définis. Par défaut, LR fixe la probabilité de coupure à 0,50 pour la classification de deux catégories.

Support Vector Machine (SVM) : a été introduit pour la première fois par Vapnik en 1995 [116] pour résoudre les problèmes de classification et de régression. L'idée de base de SVM est de dériver un hyperplan optimal qui maximise la marge entre deux classes. Une propriété intéressante de SVM est qu'il peut trouver une frontière de décision non linéaire en projetant les données à travers une fonction non linéaire ϕ dans un espace de plus grande dimension. Cela signifie que les points de données qui ne peuvent pas être séparés par une ligne droite dans leur espace d'entrée d'origine sont élevés vers un espace d'entités F où il peut y avoir un hyperplan linéaire séparant les points de données d'une classe d'une autre. Lorsque cet hyperplan serait projeté en arrière vers l'espace d'entrée I , il aurait la forme d'une courbe non linéaire. Mathématiquement, pour n échantillons de données d'apprentissage $\{(x_i, y_i)\}_{i=1}^n, x_i \in \mathbb{R}^n, y_i \in \{1, -1\}$ SVM est formulé par le problème d'optimisation suivant :

$$\text{Minimize } \phi(\omega) = \frac{1}{2}\omega^T\omega + C \sum_{i=1}^n \xi_i \quad (4.2)$$

sujet à

$$y_i(\langle \omega, \phi(x_i) \rangle + b) \geq 1 - \xi_i, i = 1, \dots, n \quad \xi_i \geq 0, i = 1, \dots, n \quad (4.3)$$

où la fonction de noyau ϕ mappe les points d'apprentissage x_i de l'espace d'entrée vers un espace de caractéristiques de dimension supérieure. Le paramètre de régularisation C contrôle le compromis entre l'obtention d'une faible erreur sur les données d'apprentissage et la minimisation de la norme des poids.

Decision tree (DT) : [14] est l'une des techniques d'apprentissage automatique les plus connues. Un arbre de décision est constitué de nœuds de décision et de nœuds feuilles. Chaque nœud de décision correspond à un test X sur un seul attribut des données d'entrée et a un certain nombre de branches, dont chacune gère un résultat du test X . Chaque nœud feuille représente une classe qui est le résultat d'une décision pour un cas. Le processus de construction d'un arbre de décision est fondamentalement un processus de division et de conquête. Un ensemble T de données d'apprentissage est constitué de k classes $(c_1, c_2, \dots, c_k)$. Si T ne comprend que des cas d'une seule classe, T sera une feuille. Si T ne contient pas de cas, T est une feuille et la classe associée à cette feuille sera affectée à la classe majeure de son nœud parent. Si T contient des cas de classes mixtes, un test basé sur un attribut a_i des données d'apprentissage sera effectué et sera divisé en n sous-ensembles $(T_1, T_2, \dots, T_n)$, où n est le nombre de résultats du test sur l'attribut a_i .

Le même processus de construction de l'arbre de décision est effectué de manière récursive sur chaque T_i , où j varie de 1 à n , jusqu'à ce que chaque sous-ensemble appartienne à une seule classe.

Random Forest (RF) : (ou Bagged DTs) [13] est une méthode d'apprentissage d'ensemble très populaire qui construit un certain nombre de classifieurs sur les données d'entraînement et combine toutes leurs sorties pour faire les meilleures prédictions sur les données de test [13]. Ainsi, l'algorithme RF est un algorithme de minimisation de la variance qui utilise le caractère aléatoire lors de la prise de décision partagée pour éviter le sur-ajustement des données d'apprentissage. Le classifieur RF qui agrège une famille de classifieurs $h(x|\theta_1), h(x|\theta_2), \dots, h(x|\theta_k)$. Chaque membre de la famille, $h(x|\theta)$, est un arbre de classification et k est le nombre d'arbres choisis dans un modèle vectoriel aléatoire. De plus, chaque θ_k est un vecteur de paramètres choisi au hasard. Si $D(x, y)$ désigne l'ensemble de données d'apprentissage, chaque arbre de classification de l'ensemble est construit à l'aide d'un sous-ensemble différent $D_{\theta_k}(x, y) \subset D(x, y)$ de l'ensemble de données d'apprentissage. Ainsi, $h(x|\theta_k)$ est le k ème arbre de classification qui utilise un sous-ensemble de caractéristiques $x_{\theta_k} \subset x$ pour construire un modèle de classification. Chaque arbre fonctionne alors comme des arbres de décision ordinaires : il partitionne les données en fonction de la valeur d'une caractéristique particulière (qui est sélectionnée au hasard dans le sous-ensemble), jusqu'à ce que les données soient entièrement partitionnées ou que la profondeur maximale autorisée soit atteinte.

La sortie finale y est obtenue en agrégeant les résultats ainsi :

$$y = \operatorname{argmax}_{p \in \{h(x_1), \dots, h(x_k)\}} \left\{ \sum_{j=1}^k (I(h(x|\theta_k) = p)) \right\} \quad (4.4)$$

où I désigne la fonction d'indicateur.

Gradient-Boosted Tree (GBT) : [50] est une technique d'apprentissage automatique utilisée pour l'analyse de la régression et pour les problèmes de classification statistique, où un modèle prédictif est obtenu sous la forme d'un ensemble de modèles de prédiction faibles, généralement des arbres de décision. Un modèle GBT est un ensemble de modèles d'arbres de classification qui permettent d'obtenir des résultats prédictifs grâce à des estimations progressivement améliorées. L'idée générale des algorithmes de GBT est de calculer une séquence d'arbres simples, où chaque arbre successif est construit pour les résidus de prédiction de l'arbre précédent. Ces techniques d'apprentissage automatique ont utilisé un ensemble d'apprentissage $(x_1, y_1), \dots, (x_n, y_n)$ de valeurs connues de x et des valeurs correspondantes de y le but est de trouver une approximation $\hat{F}(x)$ d'une fonction $F(x)$ qui minimise la valeur attendue de certains spécifiés.

Naïve Bayes (NB) : est un classifieur statistique bayésien simple [39]. Il est appelé "Naïve" car il suppose que toutes les variables contribuent à la classification et sont mutuellement corrélées. Cette hypothèse est appelée indépendance conditionnelle de classe [51]. Il s'agit d'une hypothèse irréaliste pour la plupart des ensembles de données, mais elle

conduit à un cadre de prédiction simple qui donne des résultats étonnamment bons dans de nombreux cas pratiques. Le classifieur Naïve Bayes est basé sur la théorie de Bayes.

Le théorème de Bayes est comme suit :

$$P(H/X) = \frac{P(X/H)P(H)}{P(X)} \quad (4.5)$$

Où,

- H : une hypothèse, telle que le tuple de données X appartient à la classe C spécifiée
- X : quelques preuves, décrire par mesure sur un ensemble d'attributs
- $P(H/X)$: la probabilité a posteriori que l'hypothèse H soit vérifiée compte tenu de la preuve X
- $P(H)$: probabilité préalable de H , indépendante de X
- $P(X/H)$: la probabilité postérieure de X conditionnée à H .

Les classifieurs bayésiens sont des classifieurs statistiques. Ils prédisent la probabilité d'appartenance à une classe, c'est-à-dire la probabilité qu'un échantillon donné appartienne à une classe particulière. Les classifieurs bayésiens ont montré une précision et une vitesse élevées lorsqu'ils sont appliqués à de grandes bases de données [51].

MultiLayer Perceptron (MLP) : est un classifieur basé sur la théorie des éléments de traitement multiples contribuant à générer une certaine sortie [59]. Dans un neurone avec des entrées n , l'addition cumulative du produit de ces entrées et des poids associés est l'entrée de la fonction d'activation de l'élément (neurone). Dans un réseau multicouche, chaque unité effectue le même processus et la fonction d'activation peut ne pas être la même pour tous les éléments du réseau. Un réseau neuronal avec une topologie à trois couches (couche d'entrée, cachée et de sortie) est généralement suffisant pour effectuer les mêmes relations d'entrée-sortie non linéaires complexes que tout réseau avec plus d'une couche cachée. Par conséquent, une topologie à trois couches est proposée et un nœud dans la couche cachée exécute une fonction de transfert comme indiqué dans 4.6.

$$z_j = \sigma^h(s_j^h) = \sigma^h\left(\sum_{i=0}^N \omega_{ji}^h x_i + \Theta_j^h\right) \quad (4.6)$$

Ici x est le signal d'entrée, ω est le poids associé, h est la couche réseau (couche cachée), j est l'unité de couche, N est le nombre de signaux d'entrée, σ^h est la fonction d'activation de la couche cachée et Θ est le biais, qui peut être utilisé comme une entrée unitaire avec une connexion de poids. De même, dans la couche de sortie, la réponse du système est donnée par 4.7.

$$y_k = \sigma^o(s_k^o) = \sigma^o\left(\sum_{j=0}^L \omega_{kj}^o z_j + \Theta_k^o\right) \quad (4.7)$$

Où z est le signal d'entrée (signal de sortie de la couche cachée), ω est le poids associé, o est la couche du réseau (couche de sortie), k est l'unité de couche, L est le nombre de signaux d'entrée et σ^o est la sortie fonction d'activation de couche. Θ est utilisé comme dans (1).

4.2.3 Approche de détection d'anomalies non supervisée

Les techniques de détection d'anomalies qui fonctionnent en mode non supervisé ne nécessitent pas de données d'apprentissage et sont donc les plus largement applicables. Les techniques de cette catégorie supposent implicitement que les instances normales soient beaucoup plus fréquentes que les anomalies. Si cette hypothèse n'est pas vraie, ces techniques souffrent d'un taux élevé de fausses alarmes.

Dans cette section, nous présentons les méthodes d'apprentissage automatique non supervisé utilisées dans le domaine de la détection d'anomalies et considérées dans ce travail. Plus précisément, nous nous intéressons aux méthodes ayant une implémentation officielle dans le cadre de calcul distribué open source Apache Spark.

K-Means (KM) : est un algorithme de clustering bien connu et largement utilisé [57]. KM est l'un des algorithmes de clustering les plus simples de l'apprentissage automatique qui peut être utilisé pour reconnaître automatiquement des groupes d'instances/-éléments/objets/points similaires dans l'apprentissage des données. L'algorithme classe les instances en un nombre prédéfini de groupes ou clusters spécifié par l'utilisateur (par exemple, supposons k clusters). La première étape importante consiste à choisir un ensemble de k instances en tant que centres de gravité (centres des clusters) au hasard, généralement en choisissant une pour chaque cluster aussi loin que possible les unes des autres. Ensuite, l'algorithme continue à lire chaque instance de l'ensemble de données et l'affecte au cluster le plus proche. Il existe certaines méthodes pour mesurer la distance entre l'instance et le centre de gravité, mais la plus populaire est la distance euclidienne. Les centres de gravité du cluster sont toujours recalculés après chaque insertion d'instance.

Bisecting K-Means (BKM) : est une combinaison de KM et de clustering hiérarchique [7]. Au lieu de partitionner les données en k clusters à chaque itération, BKM divise un cluster en deux sous-clusters à chaque étape de division (en utilisant KM) jusqu'à ce que k clusters soient obtenues. Comme BKM est basé sur KM, il conserve les mérites de KM et présente également certains avantages par rapport à KM. Premièrement, BKM est plus efficace lorsque k est grand. Pour l'algorithme KM, le calcul implique chaque point de données de l'ensemble de données et k centroïdes. D'un autre côté, à chaque étape de division de BKM, seuls les points de données d'un cluster et de deux centres de gravité sont impliqués dans le calcul. Ainsi, le temps de calcul est réduit. Deuxièmement, BKM produit des clusters de tailles similaires, tandis que KM est connue pour produire des clusters de tailles très différentes.

Gaussian Mixture Model (GMM) : est un algorithme de clustering populaire. GMM utilise une méthode probabiliste de clustering souple. Chaque cluster représente une distribution de probabilité dans un espace dimensionnel d , avec des points de données représentant des échantillons de cette distribution de probabilité. GMM suppose que

chaque point de données à regrouper est échantillonné à partir d'un mélange de distributions gaussiennes avec des paramètres inconnus. La distribution de probabilité $p(X)$ d'un nœud (représenté par un vecteur X) dans le réseau est définie comme une somme pondérée des distributions de probabilité du nœud dans tous les k clusters de composants selon l'équation 4.8. Chaque distribution de composant (désignée par $\mathcal{N}(X|\mu_k, \sum_k)$) est une gaussienne représentant un cluster.

$$p(X) = \sum_{k=1}^K \pi_k \mathcal{N}(X|\mu_k, \sum_k) \quad (4.8)$$

Dans l'équation 4.8, π_k est le coefficient de mélange pour le cluster k (un des K clusters), est la moyenne de la distribution normale pour le cluster k et $\sum_k$ est la covariance de la distribution normale pour le cluster k . Le coefficient de mélange π_k indique le degré d'association d'un nœud avec le cluster k . La moyenne et la covariance sont les paramètres qui définissent une distribution normale multivariée représentant un cluster. Pour une distribution normale à variable unique, la valeur de la covariance est remplacée par celle de la variance ou de l'écart type.

4.2.4 Approche générale

Dans cette section, nous présentons la méthodologie adoptée dans cette étude. La Figure 4.1 montre un schéma d'un pipeline d'apprentissage automatique. Les approches supervisées et non supervisées sont basées sur les mêmes composants principaux : initialisation, modélisation et performance. L'entrée du système comprend les ensembles de données d'apprentissage qui seront fournis à l'algorithme d'apprentissage automatique. La sortie est un modèle appris qui sera après utilisé avec de nouvelles données afin de générer des prédictions.

- **Initialization** : Les deux approches chargent un ensemble de données et préparent les données contenues afin de créer les modèles d'apprentissage automatique. Le processus de préparation transforme les données disponibles en données qui peuvent être utilisées pour former et évaluer des modèles d'apprentissage automatique. Il consiste à nettoyer, transformer (standardisation, normalisation, etc.) pour assurer l'uniformité. Le processus de préparation peut également inclure la sélection des fonctionnalités pertinentes. Ce processus économise du temps de formation, améliore la qualité et l'efficacité de l'ensemble de données. De plus, il améliore les résultats du modèle d'apprentissage automatique. L'approche supervisée utilise un ensemble de données étiquetées alors qu'il n'est pas étiqueté pour le cas du non supervisé.
- **Modélisation** : Le processus de modélisation divise l'ensemble de données chargé en ensembles d'entraînement et de test. Ce processus est réalisé en entraînant un classifieur donné sur un ensemble de données de formation associé à leurs étiquettes, pour le cas supervisé. Pour un algorithme non supervisé, la connaissance préalable des étiquettes n'est pas essentielle, le classifieur est construit avec le nombre approprié de clusters.

- **Performance** : L'évaluation est une étape clé de toute opération méthodologique. Après avoir construit un classifieur, nous devons avoir une vision claire de ses performances, pour nous assurer qu'il agira comme prévu plus tard. Afin d'évaluer les performances d'un classifieur, nous devons utiliser des paramètres et des informations bien définis. Les prédictions obtenues à partir de l'ensemble de test peuvent être utilisées pour déterminer les performances du classifieur entraîné.

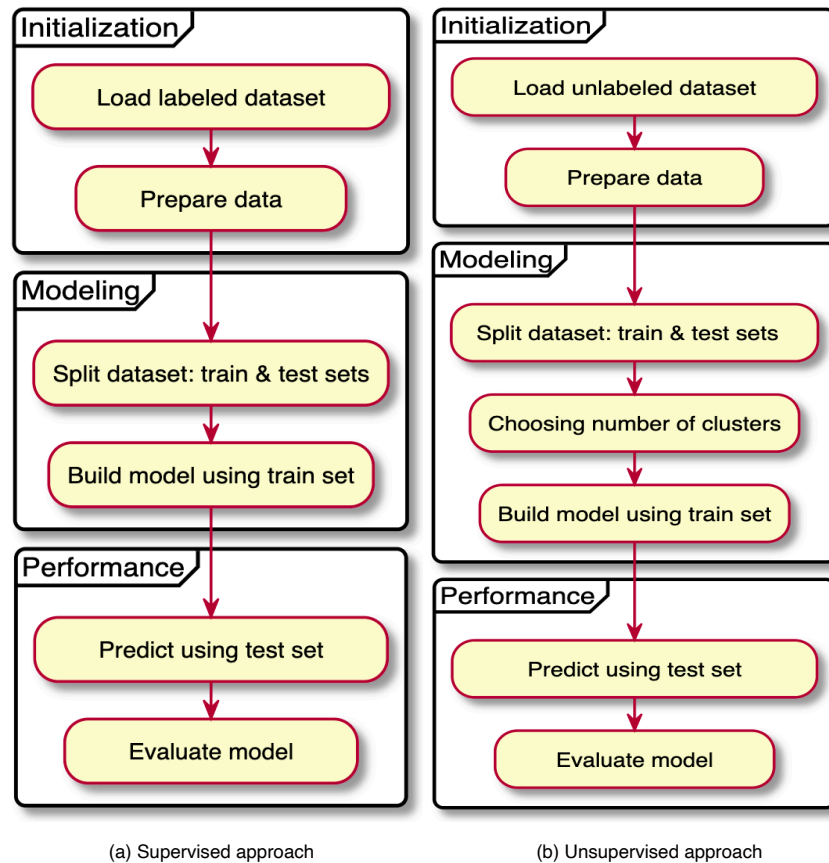


FIGURE 4.1 – Approche générale considérée

4.3 Évaluation expérimentale de performances de détection

4.3.1 Description de données

Dans cette section, nous détaillons les données de détection de fraude par carte de crédit que nous avons utilisé pour ce travail. Tout d'abord, nous tenons à souligner qu'un petit échantillon de données de Worldline contenant 284 807 transactions a été publié sur Kaggle⁶ et est, au moment de la rédaction du manuscrit, l'ensemble de données le

6. Plateforme la plus populaire pour les compétitions de science des données

plus célèbre de cette plateforme avec un score de popularité deux fois plus élevé que le deuxième⁷. Cela confirme clairement l'attractivité de la tâche de détection des fraudes et devrait être une motivation pour Wordline d'ouvrir les données au domaine public (tout en s'assurant qu'elles respectent la confidentialité et la sécurité). L'ensemble de données contient les transactions effectuées par cartes de crédit en deux jours de septembre 2013 par les titulaires de cartes européennes. Le Tableau 4.1 donne les statistiques de l'ensemble de données et montre que la classe positive, qui signifie les fraudes, représente 0,172% de toutes les transactions. Par conséquent, cet ensemble de données est très déséquilibré [31].

TABLE 4.1 – Détails de l'ensemble de données de fraude par carte de crédit Kaggle

Transactions	Classe majoritaire	Classe minoritaire	Colonnes
284 807	284 315	492	31

Il contient 31 fonctionnalités numériques. Chaque ligne de l'ensemble de données peut être représentée comme suit 4.9 :

$$V^i = (Time^i, V_1^i, V_2^i, \dots, V_{28}^i, Amount^i, Class^i) \quad (4.9)$$

Dans l'équation 4.9, les valeurs de V_1^i à V_{28}^i sont les principales composantes obtenues avec une transformation PCA en raison de problèmes de confidentialité. Ils sont numériques. Les seuls composants qui n'ont pas été transformés avec PCA sont $Time^i$, et $Amount^i$:

- $Time^i$: contient les secondes écoulées entre chaque transaction et la première transaction de l'ensemble de données ;
- $Amount^i$: est le montant de la transaction ;
- $Class^i$: représente le libellé et ne prend que 2 valeurs : la valeur 1 en cas de transaction frauduleuse et 0 dans le cas contraire.

Après avoir eu une vue de l'ensemble de données, nous effectuons une analyse exploratoire des données (EDA) pour obtenir une compréhension détaillée de cet ensemble de données.

Dans un premier temps, nous vérifions si la relation entre la variable $Time$ et la variable $Class$ existe ou non. La Figure 4.2 présente la distribution de la variable $Time$ des deux classes « transactions frauduleuses et transactions normales ». En voyant les distributions, La variable $Time$ est presque similaire pour les deux types de transactions, et donc nous pouvons déduire qu'il n'y a pas de relation entre la variable $Time$ et la variable $Class$.

Dans un second temps, nous vérifions si le montant de la transaction diffère entre les deux classes. La Figure 4.3 présente la distribution de la variable $Amount$ pour les deux

7. Lien vers l'ensemble de données Kaggle : <https://www.kaggle.com/mlg-ulb/creditcardfraud>

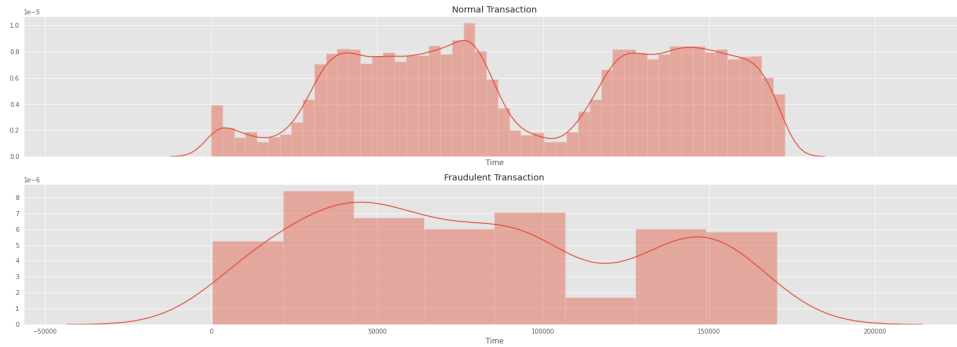


FIGURE 4.2 – La distribution de la variable Time pour les deux classes



FIGURE 4.3 – La distribution de la variable Amount pour les deux classes

classes. La plupart des transactions sont de petits montants, moins de 100. Les transactions frauduleuses ont une valeur maximale bien inférieure aux transactions normales.

Afin d'avoir une vue globale de l'ensemble de données, nous analysons la fréquence des fonctionnalités et calculons la corrélation entre elles. Sur la Figure 4.4 nous pouvons visualiser la matrice de corrélation des différentes fonctionnalités. La matrice de corrélation est un tableau représentant les coefficients de corrélation entre les groupes de fonctionnalités. Lorsque deux fonctionnalités évoluent dans le même sens, alors elles sont positivement corrélées. Sinon, si deux fonctionnalités se déplacent dans la direction opposée (une à la hausse, une à la baisse), elles sont négativement corrélées. D'après la matrice de corrélation, on peut conclure qu'aucun des composants PCA V_1 à V_{28} n'a de corrélation avec les autres. Cependant nous pouvons observer que la fonctionnalité classe a une forme de corrélations positives et négatives avec les composants V_1 à V_{28} , mais n'a aucune corrélation avec les fonctionnalités Time et Amount.

- Corrélations négatives : V_3 , V_9 , V_{10} , V_{12} , V_{14} , V_{16} , V_{17} et V_{18} sont négativement corrélés. Plus ces valeurs sont basses, plus le résultat final sera une fraude.
- Corrélations positives : V_2 , V_4 , V_{11} et V_{19} sont positivement corrélées. Plus ces valeurs sont élevées, plus le résultat final sera une transaction frauduleuse.

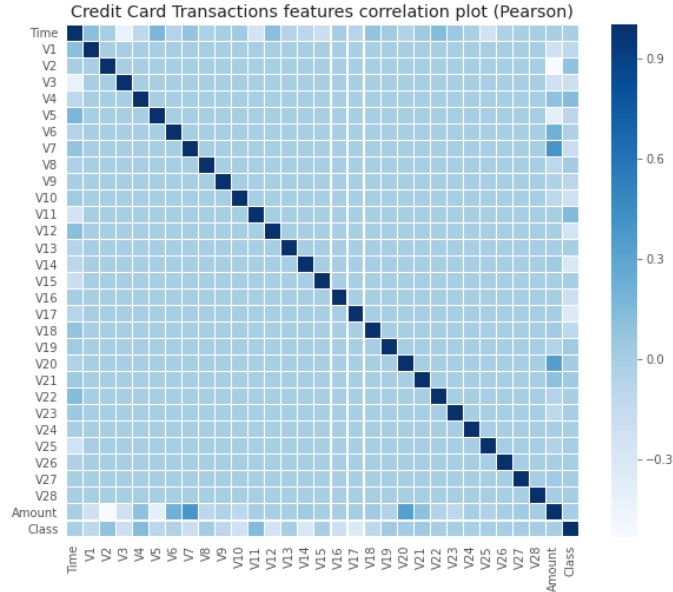


FIGURE 4.4 – Matrice de corrélation des variables de l’ensemble de données de carte de crédit

4.3.2 Protocole Expérimental

Ensembles de données de formation et de test

Les méthodes d’apprentissage automatique divisent l’ensemble de données d’entrée en ensembles d’entraînement et de test. En général, si les données ne sont pas corrélées, les ensembles d’apprentissage et de test peuvent être obtenus de manière aléatoire. Mais dans notre cas, nous connaissons le temps écoulé entre les transactions. Dès lors, nous divisons l’ensemble de données en fonction du temps d’occurrence. Comme il n’y a pas de règle empirique pour diviser un ensemble de données en ensembles d’entraînement et de test, nous considérons les deux rapports importants utilisés dans l’apprentissage automatique : 70/30 et 80/20. Le Tableau 4.2 montre que dans le cas d’une règle de 80/20, le pourcentage de classe positive de l’ensemble de données de test tombe en dessous de 20% de la fraude totale. Cela peut donc avoir un impact négatif sur les performances des méthodes étudiées. En conclusion, nous choisissons la règle 70/30 car elle donne un pourcentage de classe positive proche de 20%.

4.3.2.1 Plateforme d’apprentissage automatique Spark

Apache Spark est un moteur de traitement qui fournit à la fois un traitement en temps réel, par lots et en continu des données. Le framework Spark est conçu pour la science des données et son abstraction facilite l’analyse des données. Spark a la capacité de mettre en cache l’ensemble de données en mémoire, accélère ainsi le traitement itératif des données,

TABLE 4.2 – Statistiques des ensembles de données de formation et de test

		Total	Classe Positive
Ensemble de données		284 807 – 100%	492 – 100%
Règle 70/30	Ensemble de données de formation	199 364 – 70%	384 – 78%
	Ensemble de données de test	85 443 – 30%	108 – 22%
Règle 80/20	Ensemble de données de formation	227 845 – 80%	417 – 85%
	Ensemble de données de test	56 962 – 20%	75 – 15%

ce qui en fait un moteur de traitement idéal, en particulier pour l'apprentissage automatique. Spark est basé sur Scala, mais propose des API pour les langages de programmation Java, Python et R. Pour nos expériences, nous utilisons l'implémentation Scala.

Comme indiqué précédemment, ce travail est basé sur la bibliothèque de machine learning Spark "MLlib". Cette bibliothèque propose des algorithmes d'apprentissage courants tels que la classification, la régression, le clustering et le filtrage collaboratif. L'API Machine Learning principale pour Spark est désormais l'API basée sur DataFrame dans le package *spark.ml* qui est une évolution de l'API basée sur RDD. Selon la documentation Apache Spark : "Spark peut s'exécuter à la fois par lui-même ou sur plusieurs gestionnaires de cluster existants". Dans notre travail, nous choisissons le gestionnaire de cluster Hadoop Yarn pour le déploiement de Spark (Figure. 4.5).

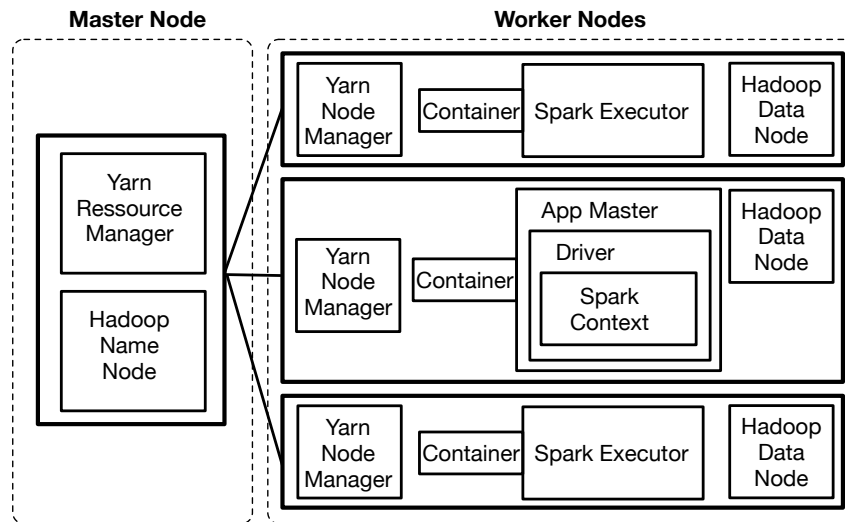


FIGURE 4.5 – Mode de cluster Hadoop Yarn pour Spark

Il existe deux modes de déploiement pour lancer des applications Spark sur Yarn : le mode cluster et le mode client. La Figure. 4.5 montre l'architecture adoptée basée sur le mode cluster où tout s'exécute à l'intérieur du cluster. En effet, le pilote Spark s'exécute dans un processus maître d'application qui est géré par Yarn sur le cluster. Dans ce mode, le client peut partir après le lancement de l'application. En mode client, le pilote s'exécute

dans le processus client et le maître d'application n'est utilisé que pour demander des ressources à Yarn. Dans nos expériences, nous exécuterons des applications basées sur des algorithmes qui peuvent prendre un temps de traitement long. Pour cette raison, le mode cluster est plus approprié. Nous pensons que le mode client est bien adapté aux travaux interactifs, mais les applications échoueront si le client s'arrête. Le mode cluster de fils nécessite une configuration d'allocation de mémoire appropriée.

Comme le montre la Figure. 4.5, notre plateforme expérimentale contient un noeud maître et trois noeuds worker configurés comme suit :

- Système d'exploitation : ubuntu-16.04.3-server-amd64
- Disque dur : 500GB SSD type
- Processeurs : 16GB RAM (4 sockets, 4 cores)
- Réseau : 2x10GB SFP+

4.3.3 Analyse des Résultats

Métriques de performance

La bibliothèque Spark "MLlib" fournit une série de métriques pour évaluer la prédiction des modèles résultants. Dans notre cas, le défi majeur est de s'attaquer au problème du déséquilibre, car les transactions légitimes sont beaucoup plus nombreuses que les transactions frauduleuses (moins de 1% du total des transactions). Ce problème conduit souvent à une exactitude extrêmement élevée (Définition 3.1) où un modèle peut atteindre jusqu'à 99% de l'exactitude de prédiction, ignorant le 1% des cas de classe minoritaire. En d'autres termes, l'exactitude ne reflète pas la réalité dans ce cas de déséquilibre des données. Pour cette raison, nous utiliserons également la métrique AUC (Définition 3.9) pour mesurer si un modèle est capable de distinguer les deux classes. Cette métrique est considérée comme une mesure de performance plus robuste, en particulier dans le cas de l'étude des ensembles déséquilibrés [126].

Résultats des méthodes ML en tenant compte de toutes les fonctionnalités

Dans notre étude de cas générale, les données ont été divisées en 70% de données de formation et 30% de test. Ainsi, comme indiqué dans le Tableau 4.2, les données de formation contiennent 78% de transactions frauduleuses, et 22% pour les données de test. Nous commençons nos expériences en étudiant le comportement de nos deux métriques choisies dans le cas général. Ce cas concerne également toutes les fonctionnalités de données (Figure. 4.6).

Afin de mieux situer les résultats, nous commençons par rendre compte de l'exactitude des méthodes supervisées. Dans la Figure 4.7, nous voyons que l'exactitude des algorithmes supervisés est supérieure à 99%. Lorsqu'il s'agit d'un déséquilibre de données aussi grave, nous devons être prudents lors de la mesure des performances du modèle. Comme il n'y a qu'une poignée d'instances frauduleuses, un modèle qui prédit que chaque exemple

	Accuracy / AUROC	
	All Features	
	Class1 Repartition bewteen train and test datasets	
	Train 78%	Test 22%
ML Methods		
Supervised	SAF - 78-22	
Unsupervised	USAF - 78-22	

FIGURE 4.6 – Étude de cas générale : toutes les métriques et toutes les fonctionnalités

appartient à la classe négative atteindra déjà plus de 99% d'exactitude. Par conséquent, cela ne nous aide pas à trouver des cas frauduleux.

Concernant les valeurs AUC des modèles supervisés, Figure 4.7, nous pouvons voir que les deux algorithmes Support Vector Machines (SVM) et Naïve Bayes (NB) ont les performances les plus faibles, ce qui signifie que ces modèles n'ont pas de capacité de séparation de classe. Les modèles de Logistic Regression (LR), Decision tree (DT) et Gradient-Boosted Tree (GBT) obtiennent un résultat moyennement bien mais pas le meilleur, suivis par Random Forest (RF) et MultiLayer Perceptron (MLP). Enfin, nous pouvons souligner que les modèles RF et MLP obtiennent les meilleurs résultats et offrent la meilleure capacité de séparation de classe.

La Figure. 4.8 montre les résultats obtenus par des modèles non supervisés. Du point de vue de la mesure de l'exactitude, Bisecting K-Means (BKM) obtient les pires performances, tandis que K-Means (KM) obtient la meilleure exactitude. En général, le score AUC est faible pour les algorithmes non supervisés.

Enfin, nous pouvons dire peut dire que dans ce cas général, les modèles supervisés donnent des résultats plus intéressants que les modèles non supervisés.

Sélection des fonctionnalités pertinentes

Dans le reste de nos expériences, nous étudions l'impact de la distribution des données de la classe minoritaire sur les performances des algorithmes d'apprentissage. Mais pour cela, nous ne nous intéresserons qu'aux fonctionnalités de données capables de faire une séparation entre les deux classes (positive, négative). Formellement, nous sélectionnons un sous-ensemble de fonctionnalités ou d'attributs dans l'ensemble de fonctionnalités et éliminons les fonctionnalités redondantes qui ne contribuent pas aux performances. Une fonctionnalité est dite importante lorsque ses distributions de données de deux classes sont divergentes (Définition 4.9). Par conséquent, cette fonctionnalité peut potentiellement séparer les deux classes et donc améliorer les performances de prédiction.

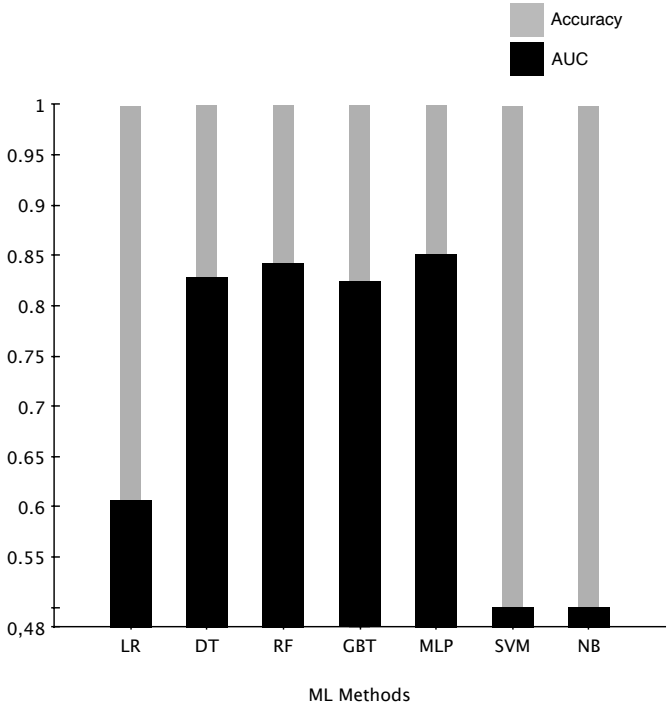


FIGURE 4.7 – SAF-78-22

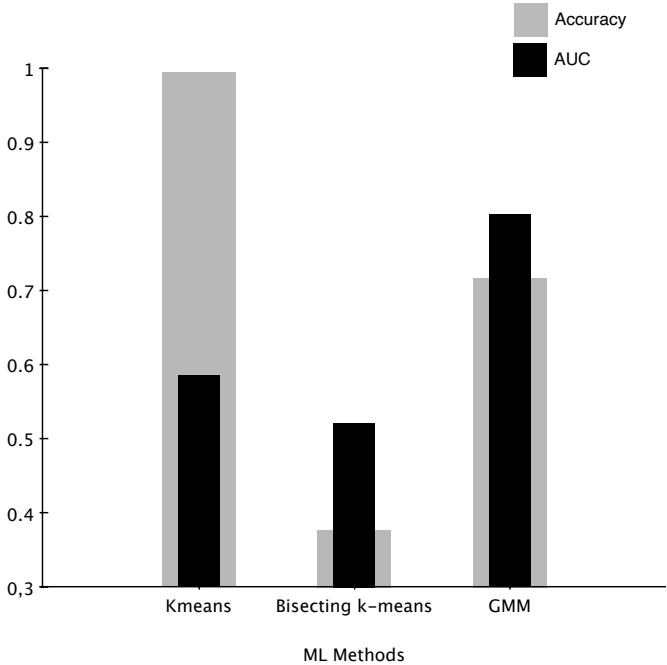


FIGURE 4.8 – USAF-78-22

Prenons l'exemple (a) illustré sur la Figure. 4.9 montrant les distributions de don-

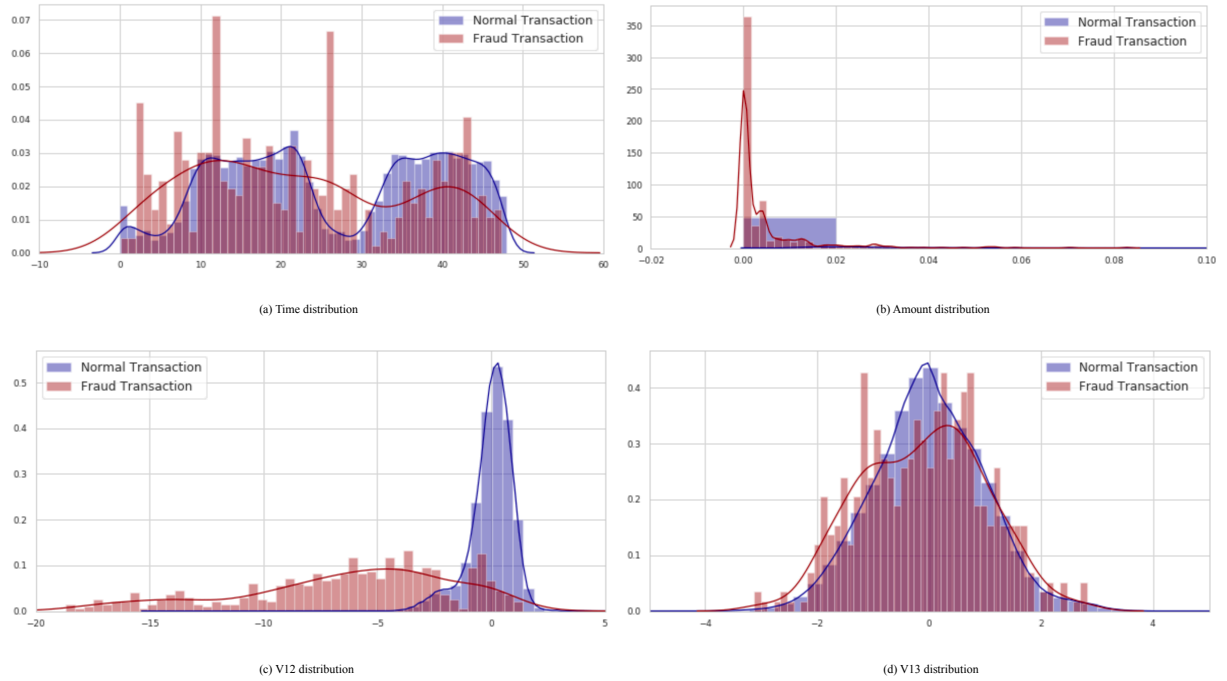


FIGURE 4.9 – Sélection des fonctionnalités pertinentes

nées des deux classes pour la fonctionnalité *Time*. Nous pouvons voir que la distribution des transactions normales (classe positive) correspond à la distribution des transactions frauduleuses (classe négative). Cela signifie que la fonctionnalité *Time* ne peut pas contribuer efficacement à la séparation entre les deux classes. De même, pour la fonctionnalité *Amount* interprétée dans la Figure 4.9.(b), nous pouvons observer que les distributions de données des deux classes sont convergentes. Mêmes conclusions pour la fonctionnalité V_{13}^i . Par conséquent, ces trois fonctionnalités ne sont pas pertinentes et elles seront éliminées lors de la construction des modèles. La Figure 4.9.(c) montre les distributions de données des deux classes pour la fonctionnalité V_{12}^i . Nous pouvons voir une divergence significative de deux distributions, c'est une fonctionnalité à fort pouvoir prédictif, nous pouvons donc la conserver lors de la construction des modèles. Enfin, les fonctionnalités importantes qui seront prises en compte dans les expériences suivantes sont : $V_3^i, V_4^i, V_9^i, V_{10}^i, V_{11}^i, V_{12}^i, V_{14}^i, V_{16}^i, V_{17}^i, V_{18}^i, V_{19}^i$.

Étude du taux de déséquilibre

Différentes expériences ont été menées pour étudier l'effet des taux de déséquilibre sur la performance des modèles supervisés et non supervisés. Les figures 4.10 et 4.13 montrent les différents paramètres considérés pour ces expériences. Tout d'abord, nous étudions séparément les méthodes supervisées et non supervisées. Les expériences ne prennent en compte que les fonctionnalités pertinentes. Pour la classe positive (transactions frauduleuses), nous avons étudié 4 divisions différentes pour les ensembles de données de formation et de test. Les prédictions obtenues à partir de ces expériences sont évaluées en termes d'exactitude et AUC, et présentées dans les Figures 4.11, 4.12, 4.14, et 4.15.

	ML Methods : Supervised							
	Relevant Features							
	Class1 Repartition bewteen train and test datasets							
	Train 80%	Test 20%	Train 60%	Test 40%	Train 40%	Test 60%	Train 20%	Test 80%
Metric								
Accuracy	Acc-RF-Sup							
AUROC	Au-RF-Sup							

FIGURE 4.10 – Eval-General-4

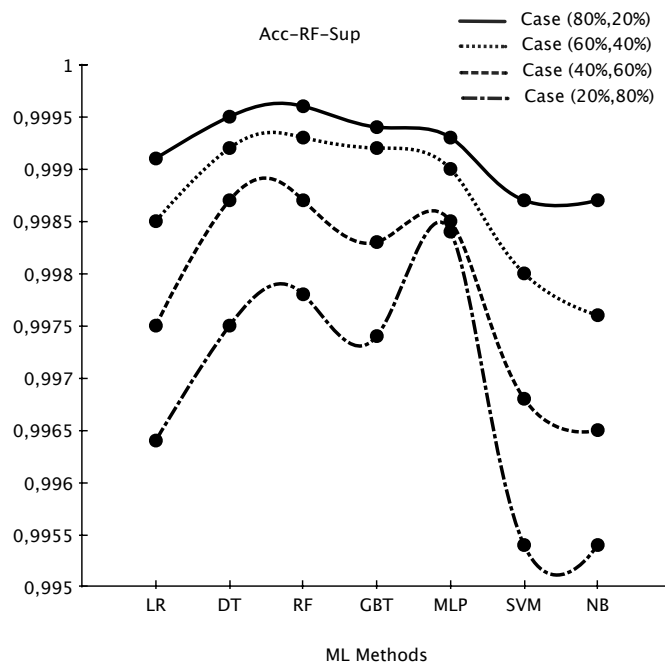


FIGURE 4.11 – Acc-RF-Sup

À partir des Figures 4.11 et 4.12, nous pouvons observer que lorsque le taux de déséquilibre augmente dans l'ensemble de formation, l'exactitude et l'AUC diminuent considérablement, et la distribution habituelle de nos données, cas (80%, 20%) avec les fonctionnalités pertinentes, est la meilleure à utiliser pour la formation des modèles supervisés de cet ensemble de données.

Les Figures 4.14 et 4.15 montrent les résultats d'expériences obtenus par les modèles non supervisés. Nous pouvons observer que la distribution du cas de données (60%, 40%) avec les fonctionnalités pertinentes est la meilleure à utiliser pour entraîner les modèles non supervisés.

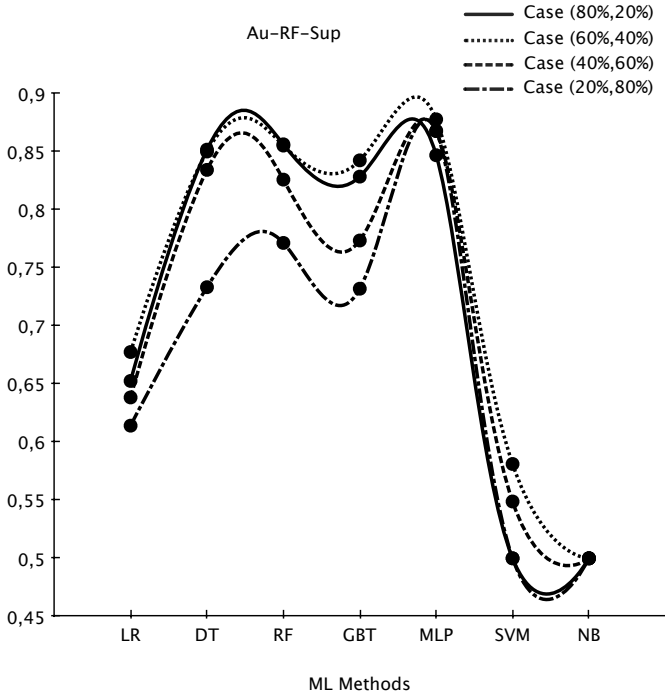


FIGURE 4.12 – Au-RF-Sup

ML Methods : Unsupervised								
Relevant Features								
Class1 Repartition bewteen train and test datasets								
Metric	Train	Test	Train	Test	Train	Test	Train	Test
	80%	20%	60%	40%	40%	60%	20%	80%
Accuracy	Acc-RF-Uns							
AUROC	Au-RF-Uns							

FIGURE 4.13 – Eval-General-5

Dernièrement, nous pouvons conclure que les modèles supervisés donnent des résultats plus intéressants que les modèles non supervisés. En revanche, nous soulignons que les modèles RF et MLP obtiennent les meilleurs résultats et offrent la meilleure capacité de séparation de classe.

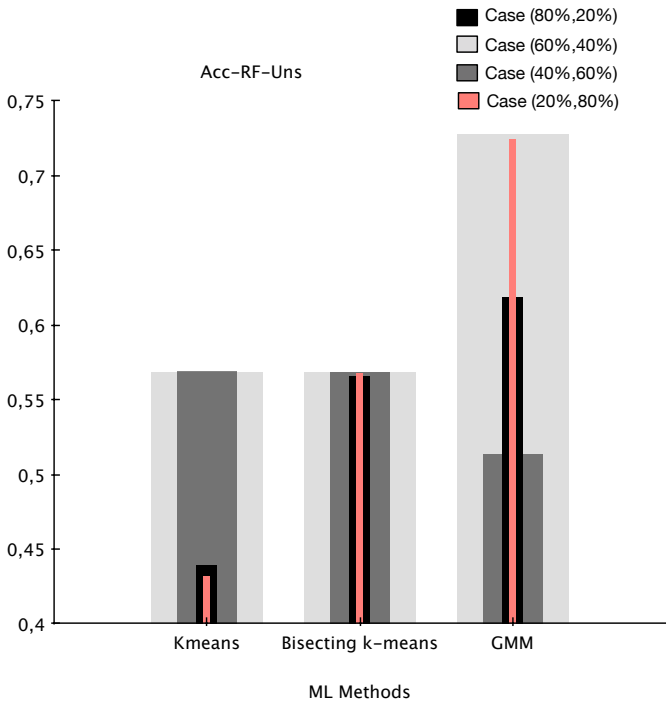


FIGURE 4.14 – Acc-RF-Uns

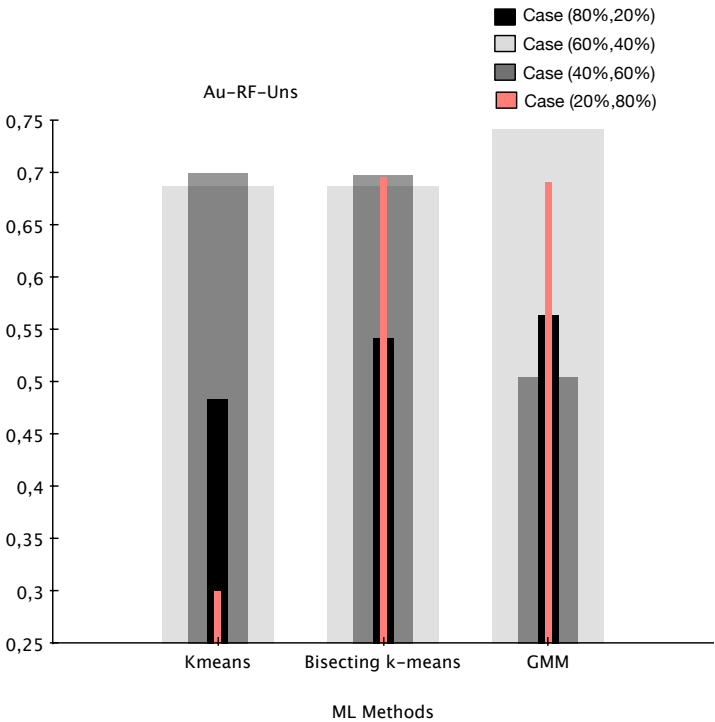


FIGURE 4.15 – Au-RF-Uns

4.4 Conclusion

La détection d'anomalies et des événements anormaux devient de plus en plus une tâche vitale dans de nombreux domaines. Ces événements peuvent souvent indiquer des activités illégales telles que la fraude, le risque, les erreurs, et l'intrusion dans le réseau, qui peuvent entraîner des résultats indésirables tels que des interruptions de service et d'autres pannes. L'apprentissage automatique est considéré comme l'une des techniques les plus efficaces pour la détection d'anomalies. Dans ce chapitre nous avons traité un problème majeur rencontré par les méthodes d'apprentissage automatique dans la détection d'anomalies. Ce problème concerne le déséquilibre des classes, les anomalies sont souvent des événements rares par rapport aux instances normales. Ce déséquilibre perturbe les algorithmes d'apprentissage, car généralement ils présentent une grande précision sur les objets normaux et une faible précision sur les objets anormaux.

Dans ce chapitre, nous avons étudié les modèles d'apprentissage automatique de manière supervisé et non supervisé pour la détection et nous avons évalué leur efficacité et comparer leurs performances sur un ensemble de données déséquilibré. Nous avons mené une étude expérimentale en utilisant diverses approches supervisées et non supervisées. Plus précisément, sept modèles de classification supervisés, à savoir, régression logistique (LR), Support Vector Machines (SVM), Decision Tree (DT), Random Forest (RF), Gradient-Boosted Tree (GBT), Naïve Bayes (NB) et Multilayer Perceptron (MLP) ainsi que 3 modèles de détection d'anomalies non supervisés, à savoir, K-Means (KM), Bisecting K-Means (BKM), et Gaussian Mixture Model (GMM), sont explorés dans cette étude. Nous avons formé tous ces modèles sur un ensemble de données de transaction par carte de crédit publique du site Web de Kaggle, qui contient 492 fraudes sur 284 807 transactions. Les étiquettes des transactions sont utilisées uniquement pour les modèles d'apprentissage supervisé. Cette étude expérimentale a été très efficace pour comprendre les véritables faiblesses des solutions existantes concernant le problème de déséquilibre de classe. Nous avons montré l'impact du déséquilibre des données sur la performance de ces approches. Ainsi, à partir des résultats obtenus, on note l'existence d'un ensemble de paramètres adaptés aux modèles supervisés et d'autres aux modèles non supervisés. Enfin, nous recommandons d'utiliser les modèles Random Forest (RF) et Multilayer Perceptron (MLP) pour obtenir des bonnes performances de prédiction.

Dans le chapitre suivant, nous introduisons notre approche OSBNR (One Side Behavioral Noise Reduction) de pré-traitement de données, basée sur le clustering et le ré-échantillonnage afin de traiter le problème du déséquilibre de classe.

Chapitre 5

Une approche basée sur l'apprentissage automatique pour traiter le problème de déséquilibre de classes en présence de bruit comportemental.

Sommaire

5.1	Introduction	88
5.2	Le principe de notre approche : OSBNR	89
5.2.1	Contexte et problématique	89
5.2.2	Clustering de données	91
5.2.3	Notre Contribution : OSBNR	93
5.3	Étude expérimentale	97
5.3.1	Protocole expérimental	97
5.3.2	Analyse des résultats	98
5.4	Conclusion	110

5.1 Introduction

Traditionnellement, les techniques d'apprentissage automatique ont été conçues pour des ensembles de données équilibrés ou qui ont certaines hypothèses sur la distribution des données. Cependant, les applications réelles sont plus souvent confrontées à des distributions de données déséquilibrées, où l'une des classes, connue sous le nom classe minoritaire, est fortement sous-représentée par rapport à l'autre classe, dite classe majoritaire [66, 31, 41]. Cette situation est connue sous le nom de problème de déséquilibre de classes [74, 47].

Le problème avec les ensembles de données déséquilibrés est que les techniques d'apprentissage automatique standard sont souvent biaisés en faveur de la classe majoritaire et, par conséquent, le taux d'erreurs de classification est plus élevé pour les instances de la classe minoritaire, qui sont généralement d'une importance cruciale comme dans le cas d'anomalies. Ainsi, une telle classification erronée des instances minoritaires a souvent de graves conséquences dans les applications réelles.

En effet certaines études affirment que le rapport de déséquilibre n'est pas la seule cause de dégradation des performances lors de l'apprentissage à partir de données déséquilibrées [104, 84, 92, 44, 43]. Ces études et d'autres soulignent que de nombreuses caractéristiques intrinsèques des données, conjuguées à une distribution de classes déséquilibrée tels que la présence de bruit, le chevauchement entre les classes, peuvent expliquer la mauvaise performance des problèmes de l'apprentissage. Dès lors, puisque la qualité des résultats obtenus par un algorithme d'apprentissage automatique étant dépendante de la qualité des données, la compréhension de ces caractéristiques intrinsèques des données, ainsi que leur relation avec le déséquilibre de classe, est crucial.

Dans ce contexte, nous nous intéressons à la détection d'anomalies survenant dans les transactions de données financières en tant que problème d'apprentissage de données très déséquilibré. La particularité du problème est que les fraudes constituant la classe minoritaire sont représentées par un très petit pourcentage de l'ensemble de données. Dans ce chapitre, nous présentons une nouvelle approche de pré-traitement de données basée sur le clustering et le ré-échantillonnage afin de traiter le problème du déséquilibre de classe avec ces caractéristiques intrinsèques des données, notamment la présence de bruit, et le chevauchement de classes. Notre méthode OSBNR (One Side Behavioral Noise Reduction) [44, 43] est basé sur deux étapes. Premièrement, une analyse de clustering est appliquée pour grouper des instances similaires de la classe minoritaire dans plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de la classe minoritaire.

Ce chapitre est structuré en deux sections. Tout d'abord, nous présentons notre approche proposée pour le traitement de problème du déséquilibre et du chevauchement de

classes. Deuxièmement, nous réalisons une série d'expériences pour valider la méthode proposée en la comparant à plusieurs méthodes de ré-échantillonnage de pointe. Enfin, nous discutons les résultats obtenus.

5.2 Le principe de notre approche : OSBNR

5.2.1 Contexte et problématique

Traditionnellement, un grand groupe de chercheurs en apprentissage automatique s'occupe de l'analyse des algorithmes d'apprentissage, pour comparer les différents algorithmes. Récemment, la question de la qualité des données a attiré l'attention de nombreux chercheurs, les incitant à identifier les propriétés des ensembles de données qui affectent les performances des classifieurs. À partir des différentes propriétés des ensembles de données comme le chevauchement entre les classes, le bruit et le déséquilibre des classes [104, 84, 92, 130, 44, 43].

Dans les problèmes à deux classes comme les systèmes de détection de fraude, la classe minoritaire (sous-représentée) est généralement appelée classe positive, alors que la classe majoritaire est considérée comme la classe négative. Ces termes sont utilisés de manière interchangeable dans la littérature. Dans ce qui suit, nous illustrerons le problème de données déséquilibrées avec un simple jeu de données synthétiques, illustré dans la Figure 5.1. Il s'agit d'un problème à deux classes avec un rapport de déséquilibre de 1 : 100, c'est-à-dire que pour chaque exemple de classe positive / minoritaire, il existe 100 exemples de classe négative / majoritaire. Les exemples positifs/minoritaires sont présentés avec un cercle rouge «o», tandis que les négatifs/majoritaires sont représentés par un cercle gris «o». On peut clairement observer que la classe positive est sous-représentée et que l'on a de sérieuses difficultés à définir une frontière de décision pour séparer les deux classes.

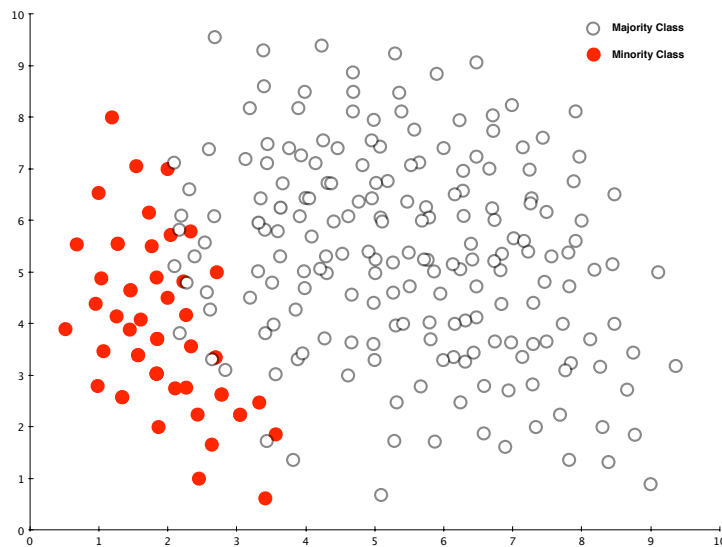


FIGURE 5.1 – Ensemble de données déséquilibré avec bruit comportemental

Différent de la notation 1 :100, une autre façon courante de se référer au degré de déséquilibre d'un problème à deux classes est le rapport de déséquilibre (IR) [128]. L'IR est défini comme le nombre d'exemples de classe négative divisé par le nombre d'exemples de classe positifs, et peut être utilisé pour trier différents ensembles de données en fonction de leur IR. IR (5.1) est défini comme :

$$IR = \frac{N_{maj}}{N_{min}} \quad (5.1)$$

où N_{maj} est la taille de l'échantillon de la classe majoritaire et N_{min} est la taille de l'échantillon de la classe minoritaire. Ainsi, dans le cas de notre exemple, l'IR est de 100. Quoi qu'il en soit, comme nous l'expliquerons plus tard, il faut tenir compte du fait que l'IR ne donne pas toujours une bonne estimation de la difficulté du jeu de données.

L'un des principaux problèmes dans les données déséquilibrées est que généralement, la classe sous-représentée est la classe d'intérêt du problème du point de vue de l'application [23, 41]. Par exemple, à la suite de la Figure 5.1, on pourrait penser que nous avons affaire à une application de détection de fraude où nous devrions différencier les transactions normales et frauduleuses en utilisant deux caractéristiques différentes. Dans ce cas, il est beaucoup plus important d'identifier correctement les transactions frauduleuses que les transactions normales, car les conséquences des transactions frauduleuses non détectées (faux positif) sont beaucoup plus graves et coûteuses que la détection d'un comportement normal de fraude (faux négatif).

De toute évidence, on voudrait une précision de 100% pour les deux classes. Cependant, la vérité est que les techniques d'apprentissage automatique sont généralement loin d'être parfaites et qu'ils ont tendance à avoir une grande précision pour la classe majoritaire tout en obtenant de mauvais résultats (plus près de 0%) pour la classe minoritaire. En conséquence, les instances de classe minoritaire sont plus souvent mal classées que celles de la classe majoritaire.

Un autre facteur critique dans les données déséquilibrées dans le monde réel concerne relativement la présence d'un nombre d'instances de la classe majoritaire située à l'intérieur de la classe minoritaire. Ce problème est communément appelé chevauchement de classes et nous l'avons appelé « bruit comportemental ». Le problème du chevauchement entre les classes apparaît lorsqu'une région de l'espace de données contient une quantité similaire de données d'apprentissage de chaque classe. Cette situation conduit à développer une inférence avec presque les mêmes probabilités a priori dans ce domaine de chevauchement, ce qui rend très difficile voire impossible la distinction entre les deux classes.

En effet, plusieurs études affirment que le rapport de déséquilibre n'est pas la seule cause de dégradation des performances lors de l'apprentissage à partir de données déséquilibrées [104, 84, 92, 130, 44, 43]. Le vrai problème est que généralement une série de difficultés qui accompagne généralement ce problème, notamment la présence de bruit ou communément appelé chevauchement de classes. En particulier, Les expériences menées

par les auteurs dans [92], à l'aide de l'algorithme l'arbre de décision, soutiennent l'affirmation que le chevauchement de classes joue un rôle important dans la dégradation des performances lorsqu'il est associé à un déséquilibre de classe, car l'ensemble de données fortement chevauché est plus affecté par le déséquilibre de classe par rapport à l'ensemble de données avec chevauchement inférieur. D'autres études ont tiré des conclusions similaires en utilisant différentes évaluations expérimentales [55].

La relation entre la performance et les problèmes de chevauchement et de déséquilibre de classe a été étudiée dans [33]. Les auteurs montrent que le chevauchement et le déséquilibre, ainsi que l'ensemble d'apprentissage, ont des effets interdépendants dans la complexité et les performances du modèle et qu'il n'est pas possible de caractériser leurs effets en les considérant uniquement isolément. Cependant, l'analyse du problème de chevauchement pris isolément montre qu'il s'agit d'un problème beaucoup plus grave que le déséquilibre.

En raison de ces difficultés, nous proposons OSBNR (One Side Behavioral Noise Reduction) [44, 43], une approche de pré-traitement de données que nous avons développée, afin de traiter le problème du déséquilibre de classe. Notre méthode OSBNR (One Side Behavioral Noise Reduction) est établie sur deux étapes. Premièrement, une analyse de clustering est appliquée pour grouper des instances similaires de la classe minoritaire dans plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de la classe minoritaire

5.2.2 Clustering de données

Le clustering de données [25, 65] fait référence au processus de regroupement de données similaires dans le même groupe ou cluster ou de fractionnement de données dissemblables en différents clusters selon certains critères prédéfinis (Figure 5.2). Étant donné un ensemble de points de données, nous pouvons utiliser un algorithme de clustering pour classer chaque point de données dans un groupe spécifique. En théorie, les points de données qui sont dans le même groupe devraient avoir des propriétés et/ou des caractéristiques similaires, tandis que les points de données dans différents groupes devraient avoir des propriétés et/ou des caractéristiques très différentes. Un clustering optimal est un partitionnement qui minimise la distance intra-cluster et maximise la distance inter-cluster.

Algorithme de clustering K-Means

K-means [57] est une méthode d'apprentissage non supervisée et une technique populaire de clustering. L'algorithme K-Means se compose de deux phases distinctes. Dans la première phase, il divise un ensemble de données en un nombre k de groupes ou clusters disjoint, et calcule le k centroïde. Dans la deuxième phase, il prend chaque point vers le cluster qui a le centroïde le plus proche. Il existe différentes méthodes pour définir la distance du centroïde le plus proche et l'une des méthodes les plus utilisées est la distance

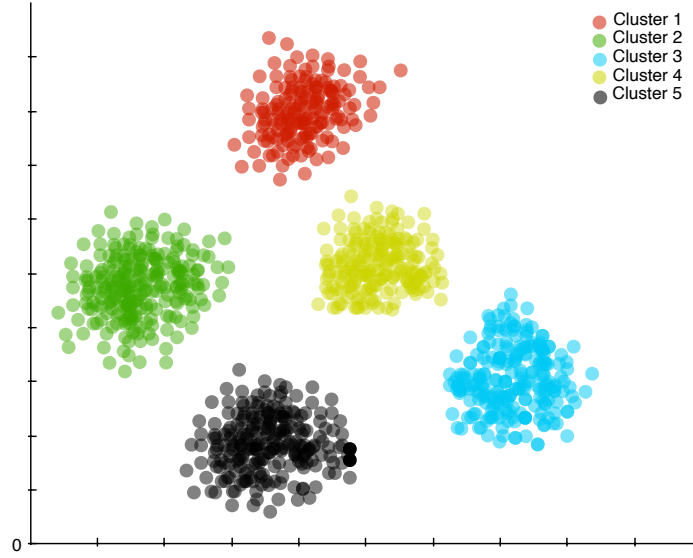


FIGURE 5.2 – Clustering sur un ensemble de points 2D, 5 clusters

euclidienne. Une fois le regroupement effectué, il recalcule le nouveau centroïde de chaque cluster et en fonction de ce centroïde, une nouvelle distance est calculée entre chaque centroïde et chaque point de données et attribue les points du cluster qui ont une distance minimale. Chaque cluster de la partition est défini par ses points membres et par centroïde. Le centroïde de chaque cluster est le point auquel la somme des distances de tous les points de ce cluster est minimisée. Donc K-Means est un algorithme itératif dans lequel il minimise la somme des distances de chaque point à son centroïde de cluster, sur tous les clusters.

Afin d'illustrer le fonctionnement de K-means, soit $X = x_i, i = 1, \dots, n$ un ensemble de n points de d -dimension à regrouper en K clusters, $C = c_k, k = 1, \dots, K$. L'algorithme de clustering K-Means est spécifié comme suit [57] :

Entrées : ensemble de points et le nombre de clusters K

Sortie : K clusters

1. Initialiser le nombre de cluster K et le centroïde
2. Pour chaque point x_i , calculer la distance euclidienne d , entre le centroïde c_k et le point x_i , la distance euclidienne (d_{in}) entre deux points (x_i, x_j) , avec d dimensions est calculée comme suit (5.2) :

$$d_{ij} = \sqrt{\sum_{d=1}^n (x_{ik} - x_{jk})^2} \quad (5.2)$$

3. Attribuer tous les points x_i au centroïde le plus proche en fonction de la distance d .

4. Une fois que tous les points ont été attribués, recalculer la nouvelle position du centroïde en utilisant la relation donnée ci-dessous (5.3) :

$$c_k = \frac{1}{k} \sum_{i=1}^n (x_i) \quad (5.3)$$

5. Arrêter si aucun point de données n'a été ré-attribué, sinon répéter les étapes 3 et 4.

La Figure 5.3 montre une illustration de l'algorithme K-Means sur un ensemble de données bidimensionnel avec trois clusters [65].

L'algorithme K-Means est l'algorithme de clustering le plus largement utilisé. Sa popularité peut être attribuée à plusieurs raisons [34]. Premièrement, il est conceptuellement simple et facile à mettre en œuvre. Pratiquement tous les logiciels d'exploration de données en incluent une implémentation. Deuxièmement, il est polyvalent, c'est-à-dire que presque tous les aspects de l'algorithme (initialisation, fonction de distance, critère de terminaison, etc.) peuvent être modifiés. Notre approche OSBNR utilise K-Means comme algorithme de clustering [57] pour diviser les instances de la classe minoritaire qui représente la fraude en clusters en fonction de leurs similitudes.

5.2.3 Notre Contribution : OSBNR

Dans cette thèse, nous nous intéressons au problème de détection de fraude par carte bancaire. Il s'agit d'un bon exemple du problème de déséquilibre et de chevauchement de classes [31, 44, 43]. Étant donné que certains utilisateurs partagent un comportement où les transactions sont similaires. Alors que le comportement de transaction de certains utilisateurs ne ressemble à aucun comportement et peut même se comporter comme des transactions contrairement à leurs étiquettes. Par exemple, si les informations de carte de crédit d'un utilisateur sont volées, les fraudeurs effectueront plusieurs transactions importantes en peu de temps pour maximiser les avantages, tandis que certains utilisateurs normaux peuvent également effectuer plusieurs transactions importantes en peu de temps pour certaines raisons. Le comportement normal d'un utilisateur individuel est donc proche de la fraude. Nous définissons ce type de transaction comme un bruit de comportement. L'existence de ce bruit de comportement pousse le système à juger certaines transactions frauduleuses comme des transactions authentiques, ce qui conduit à une classification erronée qui peut être coûteuse. Les transactions frauduleuses non détectées (faux positif) sont beaucoup plus graves et coûteuses que la détection d'un comportement normal de fraude (faux négatif). Le coût des faux positifs est de nature financière, il varie en fonction du montant de la transaction. En revanche, le coût des faux négatifs se mesure en termes d'insatisfaction client, et ce dernier peut être résolu par des stratégies de compensation et de fidélisation des clients.

L'objectif principal de notre approche proposée OSBNR est de traiter le problème de déséquilibre et de chevauchement de classes pour améliorer la classification des classes minoritaires. OSBNR consiste principalement sur deux étapes. Tout d'abord, un clustering

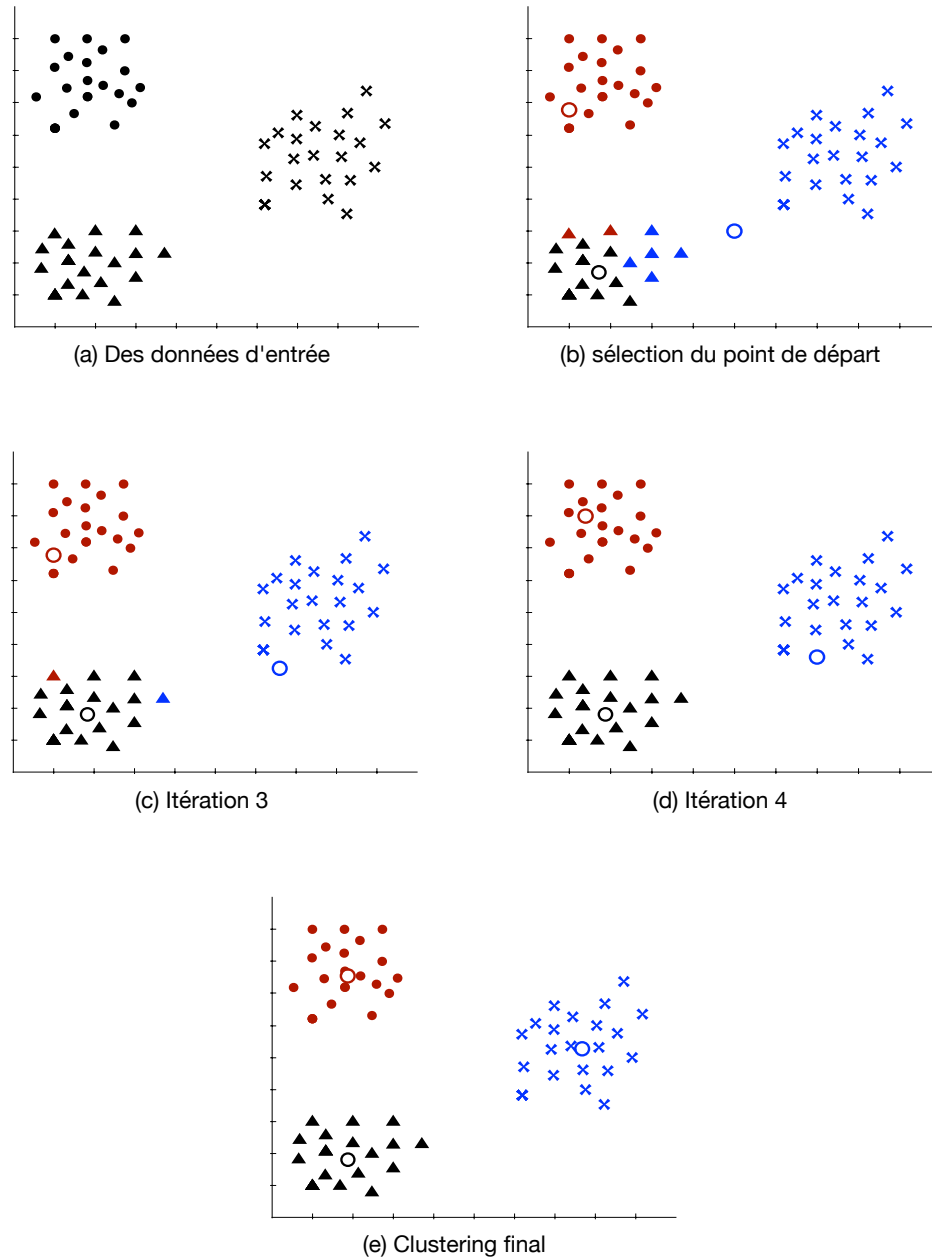


FIGURE 5.3 – Illustration de l’algorithme K-Means. (a) Données d’entrée bidimensionnelles avec trois groupes ; (b) trois points de départ sélectionnés comme centres de cluster et attribution initiale des points de données aux clusters ; (c) et (d) des itérations intermédiaires mettant à jour les étiquettes de cluster et leurs centres ; (e) regroupement final obtenu par l’algorithme K-Means à la convergence.

est appliqué pour grouper les instances similaires de la classe minoritaire contenant les transactions frauduleuses dans plusieurs sous-ensembles, qui forment plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire (transactions normales), considérées comme du bruit comportemental, qui se chevauchent avec les comportements de transactions frauduleuses.

Le processus général de notre approche est illustré dans la Figure 5.4.

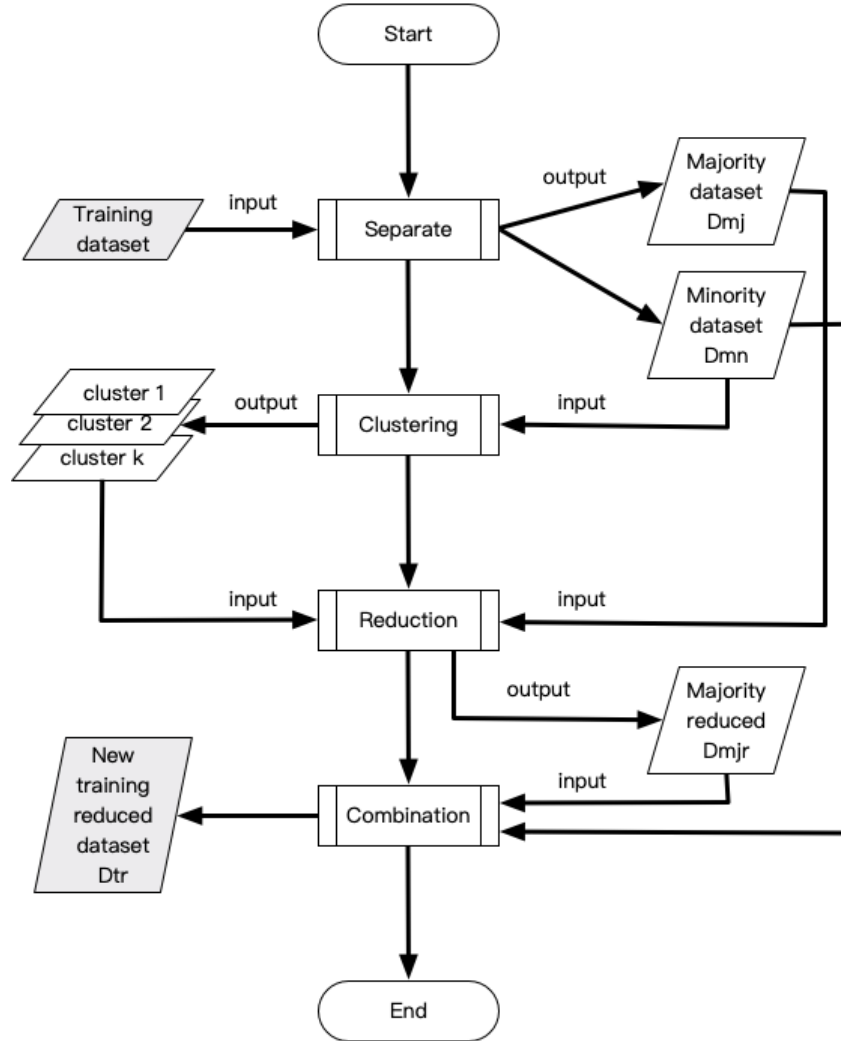


FIGURE 5.4 – Flowchart de l'approche OSBNR

Soit $\mathcal{A}_0 = \{x_i \in \mathbb{R}^n, i = 1, \dots, n\}$ l'ensemble d'apprentissage, ou $x_i = \{x_1, x_2, \dots, x_n\}$. Chaque observation x_i dispose d'une étiquette $\text{label}(x_i)$ positive (+, données minoritaires) ou négative (-, données majoritaires). On notera par la suite $\mathcal{A}^+$ l'ensemble des données positives et par $\mathcal{A}^-$ l'ensemble des données négatives.

Les itérations de notre approche sont résumées dans l'algorithme 1.

Les phases principales de notre approche proposée OSBNR :

Require: $\mathcal{A}_0$: The training set ; K : The number of clusters ;
Ensure: $\mathcal{A}_{final}$: The new training set without behavioral noise ;

- 1: Separate $\mathcal{A}_0$ into $\mathcal{A}_0^-$ (the set of majority instances) and $\mathcal{A}_0^+$ (the set of minority instances) ;
- 2: $N^- = \mathcal{A}_0^-.sum()$ (the number of majority instances) ;
- 3: **Step 1** : Clustering Analysis
- 4: **for** $k = 1$ to K **do**
- 5: $C_k = getCluster(\mathcal{A}_0^+, k)$ (Minority cluster) ;
- 6: $c_k = getCentroid(C_k, k)$ (Centroid of minority cluster) ;
- 7: **end for**
- 8: **Step 2** : Behavioral noise Reduction
- 9: **for** $k = 1$ to K **do**
- 10: $Seuil_k = \max EuclideanDistance(C_k, c_k)$;
- 11: **for** $j = 1$ to N^- **do**
- 12: $d_{kj} = EuclideanDistance(\mathcal{A}_0^-[j], c_k)$;
- 13: **if** $Seuil_k \geq d_{kj}$ **then**
- 14: $Noisydata = \mathcal{A}_0^-[j]$;
- 15: **end if**
- 16: **end for**
- 17: **end for**
- 18: $\mathcal{A}_{final}^- = \mathcal{A}_0^- \text{Except}(Noisydata)$;
- 19: $\mathcal{A}_{final} = \mathcal{A}_{final}^- \text{Union}(\mathcal{A}_0^+)$;
- 20: **return** $\mathcal{A}_{final}$;

Algorithm 1: One Side Behavioral Noise Reduction (OSBNR)

- **Entrées** : Ensemble d'apprentissage , $\mathcal{A}_0 = \mathcal{A}_0^- \cup \mathcal{A}_0^+$, le paramètre K : nombre de clusters
- **Sorties** : Ensemble d'apprentissage sous échantillonnée $\mathcal{A}_{final}(\mathcal{A}_{final} \leq \mathcal{A}_0)$, telle que $\mathcal{A}_{final}^+ = \mathcal{A}_0^+$
- **Phase de séparation** : consiste à séparer l'ensemble des données positives $\mathcal{A}_0^+$ et l'ensemble des données négatives $\mathcal{A}_0^-$ de l'ensemble d'apprentissage $\mathcal{A}_0$.
- **Phase de clustering** : consiste à diviser l'ensemble des données minoritaires $\mathcal{A}_0^+$ en K clusters, en utilisant l'algorithme K-Means, ou chaque cluster semble avoir des caractéristiques distinctes dans l'espace des caractéristique de haute dimension. Ensuite, calculer la distance $D_{ik}^+(x_i, c_k)$ entre chaque instance $x_i \in \mathcal{A}_0^+, i = 1, \dots, N^+$ de l'ensemble des données minoritaires et le centroïde c_k de chaque cluster à l'aide de la fonction de distance euclidienne 5.2, pour ensuite trouver les seuils qui représentent la distance maximale du centroïde pour chaque cluster, tel que $Seuil_k = \max D_{ik}^+(x_i, c_k), i = 1, \dots, N^+$ et $k = 1, \dots, K$. (N^+ est le nombre de données positives, et N^- est le nombre de données négatives)
- **Phase de réduction** : son objectif est d'éliminer les instances majoritaires qui se chevauchent avec celles minoritaire. Cette phase consiste à calculer la distance D_{ik}^-

entre chaque instance $x_i \in \mathcal{A}_0^-, i = 1, \dots, N^-$ de l'ensemble des données majoritaires et le centroïde de chaque cluster $c_k \in \mathcal{A}_o^+$ de l'ensemble des données minoritaires à l'aide de la fonction de distance euclidienne. pour chaque instance $x_i \in \mathcal{A}_0^-$, si $D_{ik}^-(x_i, c_k) \leq Seuil_k$, alors $\mathcal{A}_{final}^- \leftarrow \mathcal{A}_o^- - \{x_i\}$. L'identification et l'élimination précises de ces instances maximisent la visibilité des instances de classe minoritaire.

- **Phase de combinaison** : consiste à combiné l'ensemble de données majoritaires réduit $\mathcal{A}_{final}^+$ avec l'ensemble de données minoritaires $\mathcal{A}_0^+$ pour avoir un nouvel ensemble de données d'apprentissage $\mathcal{A}_{final}$.

5.3 Étude expérimentale

Dans cette section, nous fournissons une évaluation expérimentale en utilisant le même ensemble de données utilisé dans l'étude expérimentale précédente (section 4.3.1). Afin d'étudier l'efficacité de la méthode proposée, nous la comparons avec plusieurs méthodes d'échantillonnage de pointe capables de traiter des données déséquilibrées.

5.3.1 Protocole expérimental

Dans cette section, nous décrivons les différentes parties de notre protocole expérimental pour la détection de fraude par carte de crédit.

Modèles ML et techniques de ré-échantillonnage

Dans cette étude expérimentale, nous avons appliqué diverses techniques de rééchantillonnage telles que Condensed Nearest Neighbour Rule (CNN) [56], Tomek's links (TL) [113], One-Side Selection (OSS) [77], Edited Nearest Neighbour Rule (ENN) [121], Repeated Edited Nearest Neighbor (RENN) [112], All-KNN (AKNN) [112] et Neighbor Cleaning Rule (NCR) [79]. Puis nous avons évalué leurs performances avec l'approche proposée OSBNR en utilisant les meilleurs classifieurs qui sont largement utilisés dans la détection de fraude : Random Forest (RF) et MultiLayer Perceptron (MLP) [41, 11, 29, 32]

Les paramètres des modèles RF et MLP ont été déterminés à partir de divers tests préliminaires effectués sur les données d'apprentissage, comme indiqué dans le Tableau 5.1.

Mesures d'évaluation

Les mesures d'évaluation jouent un rôle important pour évaluer et guider les algorithmes d'apprentissage [108]. La mesure couramment utilisée est l'exactitude. Cependant, l'exactitude (3.1) n'est pas un bon indicateur de la performance réelle de la classification lorsque la distribution des classes n'est pas uniforme, en particulier pour la classe positive (minoritaire). En effet, parce qu'elle a moins d'effet sur l'exactitude que la classe négative (majoritaire). Comme dans [93], nous considérons d'autres métriques : Précision or

TABLE 5.1 – Paramètres RF et MLP utilisés

Classifieurs	Paramètres
Random Forest (RF)	Nombre d'arbres = 20
	Profondeur de chaque arbre = 8
	Impureté = Gini
MultiLayer Perceptron (MLP)	Nombre d'itérations = 100
	Paramètre de tolérance = 1e-6

Positive Predictive Value (3.7), True Positive Rate (3.6), F-mesure ou F1-score (3.8), AUC (3.9).

5.3.2 Analyse des résultats

Ensembles de données d'apprentissage et de test utilisés pour les expériences

Nous présentons différentes expériences pour comparer les performances de notre approche OSBNR proposée et les méthodes de ré-échantillonnage de pointe (CNN, ENN, AKNN, RENN, TL, OSS et NCR). Comme il n'y a pas de règle de base pour diviser un ensemble de données en ensembles d'apprentissage et de test, nous avons remarqué que dans le cas de la règle 70/30, les pourcentages de la classe minoritaire des ensembles d'apprentissage et de test sont : 80%, 20% respectivement de la fraude totale. Afin de démontrer l'efficacité de la méthode OSBNR proposée pour traiter le problème du déséquilibre et du chevauchement entre les classes, nous avons étudié 3 divisions différentes de l'ensemble de données en ré-échantillonnant la classe minoritaire sur la base des ratios : 80/20, 70/30, 60/40 et la classe majoritaire basée sur le ratio 70/30. Le Tableau 5.2 présente les ensembles de données d'apprentissage et de test utilisés comme entrée de notre approche OSBNR (Figure 5.4).

TABLE 5.2 – Ensembles de formation et de test utilisés pour l'expérimentation

		Totale	Classe majoritaire	Classe minoritaire
100%	Ensemble de données	284 807	284315 - 100%	492 - 100%
Règle 80/20	Formation	199 413	199020 - 70%	393 - 80%
	Test	85 394	85295 - 30%	99 - 20%
Règle 70/30	Formation	199 364	199020 - 70%	344 - 70%
	Test	85 443	85295 - 30%	148 - 30%
Règle 60/40	Formation	199 315	199020 - 70%	295 - 60%
	Test	85 492	85295 - 30%	197 - 40%

Étude de performance de l'OSBNR : cas de toutes les fonctionnalités

Dans cette section, nous analysons l'impact de l'approche OSBNR proposée sur les performances de chaque classifieur en le comparant aux méthodes de rééchantillonnage existantes en tenant compte de toutes les fonctionnalités et selon 3 divisions différentes de l'ensemble de données. Les résultats sont calculés selon quatre métriques : AUC, Précision, Rappel et F1-score. Les figures 5.5 et 5.6 montrent les résultats en utilisant la métrique AUC pour les classifieurs RF et MLP respectivement. L'observation la plus intéressante est que l'OSBNR proposé offre des performances nettement meilleures du point de vue AUC par rapport aux autres méthodes pour les deux classifieurs pour toutes les distributions des ensembles d'apprentissage et de test. Pour le classifieur RF, le meilleur score est obtenu par RF_OSBNR avec une valeur AUC de ($AUC = 0.9341$), RF_CNN prend la deuxième place avec un score de ($AUC = 0.9079$), lorsque la formation et le test les ensembles sont définis sur la règle de 80/20. Pour le classifieur MLP, le meilleur score AUC est obtenue par MLP_OSBNR avec une valeur de ($AUC = 0,9487$), suivi de MLP_OSS avec un score de ($AUC = 0,9079$) lorsque l'ensemble de formation et de test sont définis sur la base de règle de 80/20.

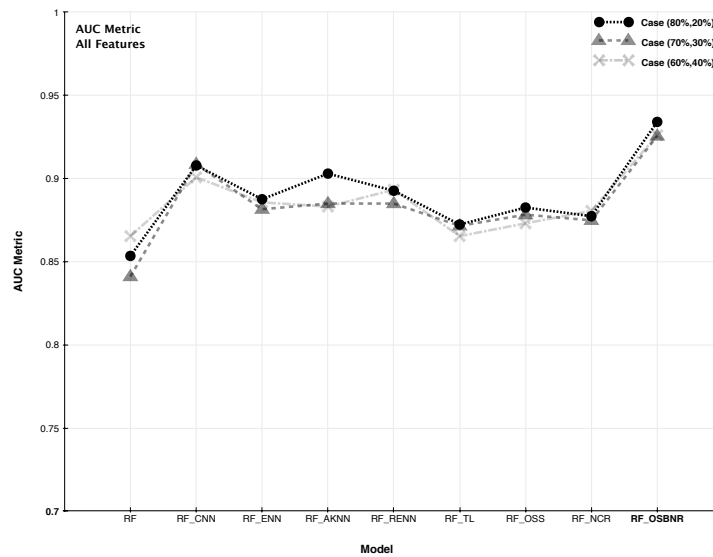


FIGURE 5.5 – AUC pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalités

De même, les Figures 5.7 et 5.8 présentent les résultats en termes de Précision. Les résultats illustrés dans la Figure 5.7 montrent que OSBNR surpasse les autres méthodes de ré-échantillonnage pour toutes les distributions des ensembles d'apprentissage et de test. Le meilleur score de précision pour le classifieur RF est obtenu par l'approche OSBNR lorsque les ensembles de formation et de test sont fixés à 80% et 20% de fraude avec un score de ($Precision = 0.8686$), tandis que RF_CNN prend la deuxième place avec un score de ($Precision = 0.8163$). Similaire pour MLP, comme illustré dans la Figure 5.8, il est clair que le meilleur score de précision est obtenu par MLP_OSBNR avec un score de ($Precision = 0.8979$), suivi de MLP_OSS avec un score de ($Precision = 0.8511$). Sur

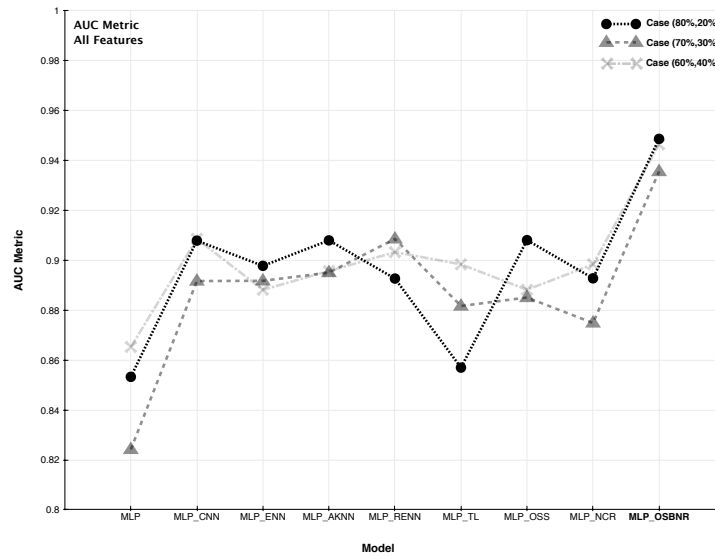


FIGURE 5.6 – AUC pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalités

la base de ces résultats, nous concluons que OSBNR peut améliorer considérablement le taux de reconnaissance des échantillons minoritaires.

Les figures 5.9 et 5.10 montrent les résultats obtenus en utilisant la mesure Rappel. L'observation la plus intéressante est que les classifieurs RF et MLP sans prétraitement offrent clairement les meilleures performances en termes de métrique de rappel. Un tel résultat était attendu en quelque sorte, car les méthodes de ré-échantillonnage ont été introduites pour gérer les problèmes de déséquilibre et de chevauchement de classe. Elles éliminent les instances de la classe majoritaire considérées comme du bruit pour améliorer la prédiction des instances de la classe minoritaire, et ce peut légèrement augmenter le taux de faux négatifs. Pour le classifieur RF, le deuxième meilleur score de rappel est obtenu par RF_OSS avec une valeur de ($Rappel = 0,96$) lorsque les ensembles de formation et de test sont définis à 60% et 40% de fraude. De même, pour MLP, la deuxième place est occupée par MLP_OSS avec un score de ($Rappel = 0,9329$) avec la règle 60/40.

Comme pour la mesure F1-score, les Figures 5.11 et 5.12 présentent les résultats de toutes les méthodes de ré-échantillonnage en appliquant les deux classifieurs d'apprentissage. Nous observons que l'approche OSBNR surpasse les autres méthodes de ré-échantillonnage pour toutes les distributions des ensembles d'apprentissage et de test pour deux classifieurs. Pour RF, le meilleur F1-score est obtenu par RF_OSBNR avec un score de ($F1 - score = 0.8572$) selon la règle 70/30, suivi par RF_AKNN avec un score de ($F1 - score = 0.8436$) lors de l'application de la règle 60/40 pour diviser les ensembles d'apprentissage et de test. En ce qui concerne MLP, le meilleur score est obtenu par MLP_CNN avec un F1-score de 0,8679 lorsque les ensembles d'apprentissage et de test sont fixés à 60% et 40% de fraude, MLP_OSBNR prend la deuxième place avec une valeur de 0,8648.

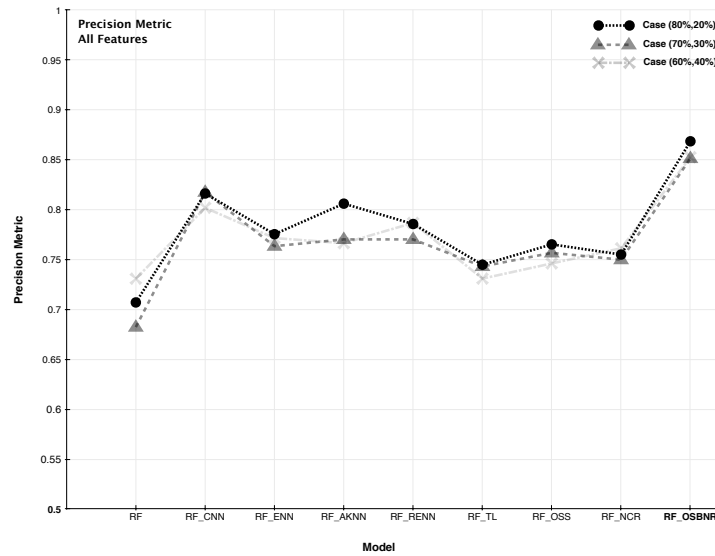


FIGURE 5.7 – Précision pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalités

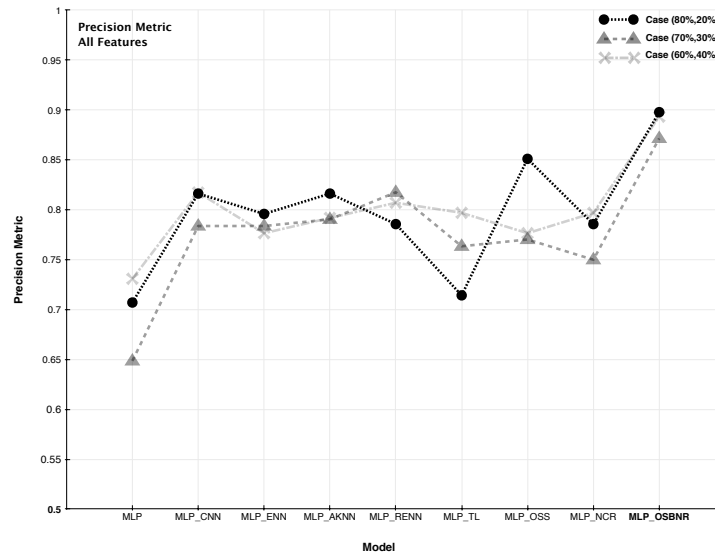


FIGURE 5.8 – Précision pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalités

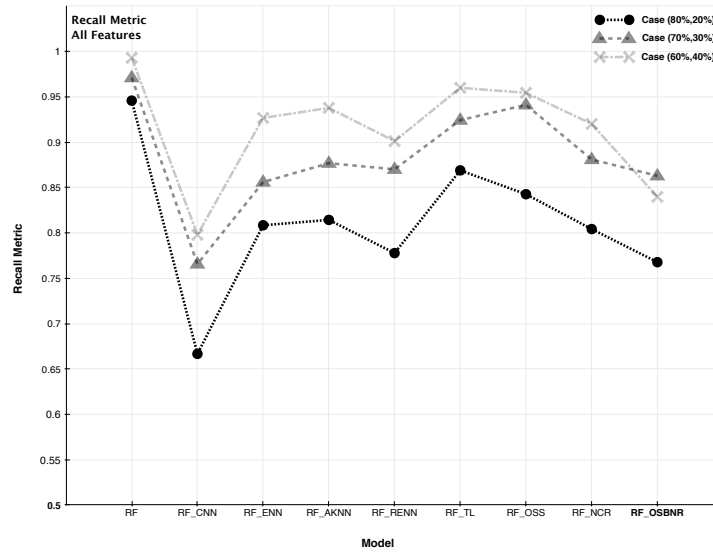


FIGURE 5.9 – Rappel pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalités

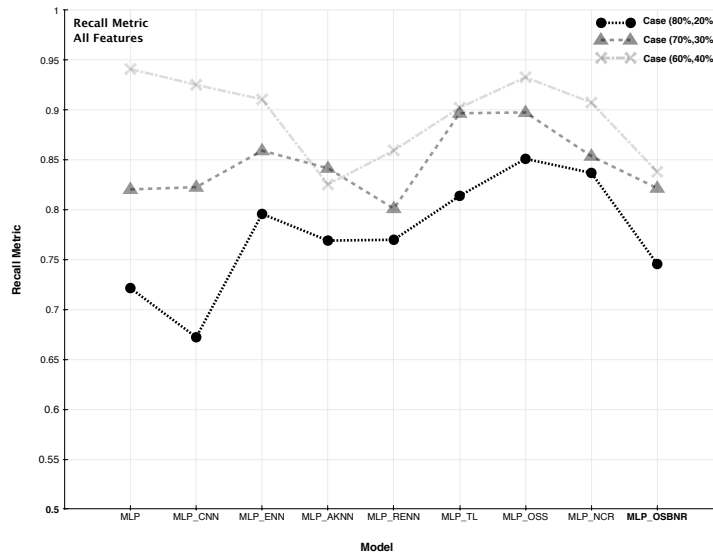


FIGURE 5.10 – Rappel pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalités

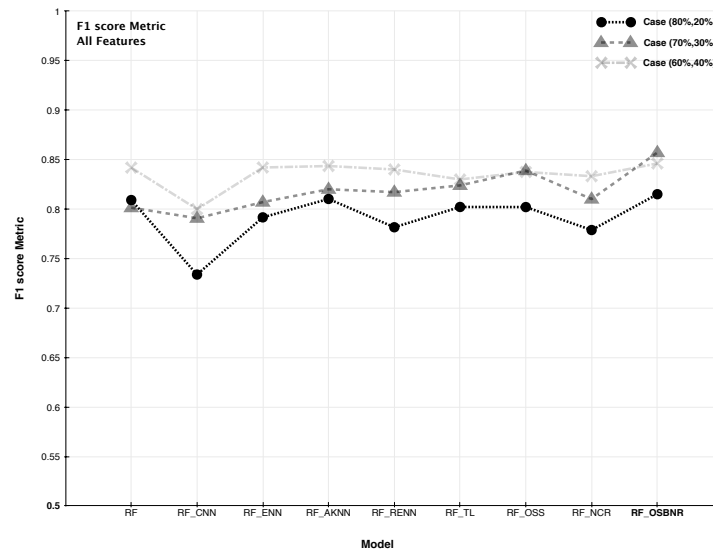


FIGURE 5.11 – F1-score pour OSBNR et les approches de ré-échantillonnage de référence : RF comme classifieur de base et cas de toutes les fonctionnalisés

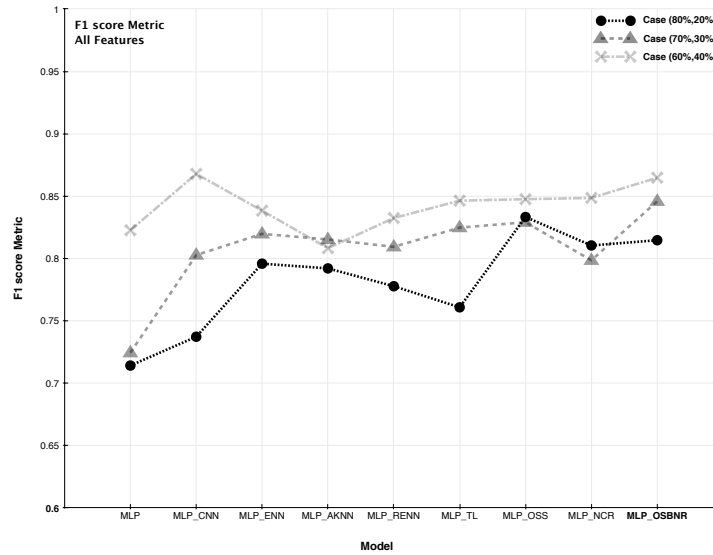


FIGURE 5.12 – F1-score pour OSBNR et les approches de ré-échantillonnage de référence : MLP comme classifieur de base et cas de toutes les fonctionnalisés

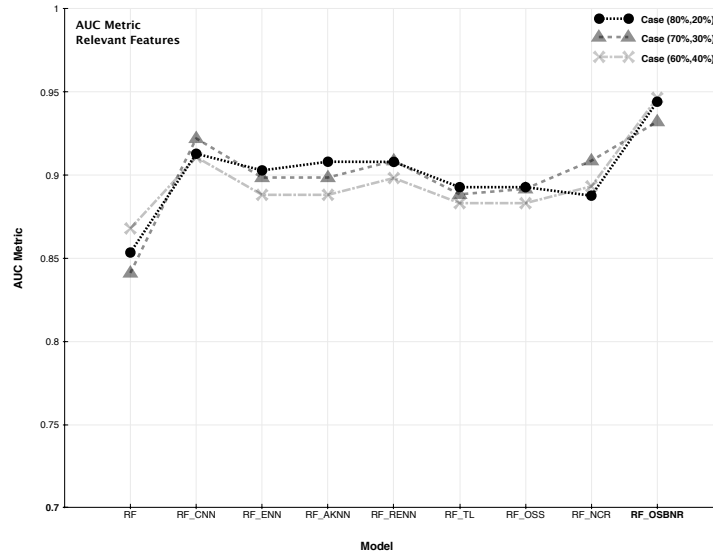


FIGURE 5.13 – AUC pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base et cas des fonctionnalités pertinentes

Étude de performance de l'OSBNR : cas de fonctionnalités pertinentes

Dans cette section, nous analysons l'impact de l'OSBNR proposé sur les performances de chaque classifieur en le comparant aux méthodes de ré-échantillonnage existantes en tenant compte des fonctionnalités pertinentes(4.3.3) et également en fonction des 3 divisions différentes de l'ensemble de données. Les résultats sont calculés pour quatre paramètres : AUC, précision, rappel et F1-score . Afin de mieux situer les résultats, nous commençons par rendre compte de la métrique AUC à partir des Figures 5.13 et 5.14. Nous pouvons voir que les meilleurs scores AUC sont obtenus par RF et MLP combinés avec OSBNR pour toutes les distributions des ensembles d'apprentissage et de test. Pour le classifieur RF, le meilleur score est obtenu par RF_OSBNR avec une valeur AUC de ($AUC = 0.9442$), RF_CNN prend la deuxième place avec un score de ($AUC = 0.9129$) selon le 80/20 règle. Concernant le classifieur MLP, le meilleur score AUC est obtenu par MLP combiné avec OSBNR lors de l'application de la règle 80/20 avec une valeur de ($AUC = 0.9543$), suivi de MLP_RENN avec un score de ($AUC = 0.9289$), tandis que MLP_OSS a obtenu un score de ($AUC = 0.9079$). À partir de ces résultats, nous pouvons conclure qu'après avoir éliminé les fonctionnalités redondantes qui ne contribuent pas à la performance, nous obtenons une amélioration des performances prédictives.

De même, les Figures 5.15 et 5.16 présentent les résultats en termes de précision. Les résultats illustrés dans la Figure 5.15 montrent que le meilleur score de précision pour le classifieur RF est obtenu par l'approche OSBNR pour toutes les distributions des ensembles d'apprentissage et de test avec un meilleur score de ($Precision = 0.8934$) selon la règle des 80/20, ce qui signifie que ce modèle offre une meilleure prédiction des instances minoritaires. Similaire en MLP, comme l'illustre la Figure 5.16, il est clair que le meilleur score de précision est obtenu par MLP_OSBNR avec un meilleur score de ($Precision = 0.909$).

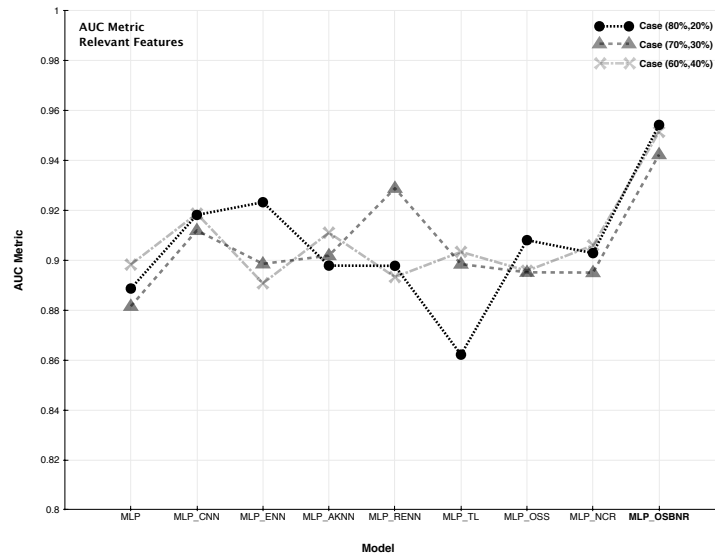


FIGURE 5.14 – AUC pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes

En ce qui concerne la mesure Rappel, les Figures 5.17 et 5.18 montrent les résultats obtenus. nous remarquons également que les classifieurs RF et MLP sans prétraitement offrent clairement les meilleures performances en termes de métrique de rappel. Pour le classifieur RF, le deuxième meilleur score de rappel est obtenu par RF_OSS avec une valeur de ($Rappel = 0,928$) lorsque les ensembles de formation et de test sont définis à 70% et 30% de fraude. De même, pour MLP, la deuxième place est occupée par MLP_AKNN avec un score de ($Rappel = 0,9474$) selon la règle des 70/30.

Quant à la mesure F1-score, les Figures 5.17 et 5.18 montrent les résultats de toutes les méthodes de ré-échantillonnage en appliquant les deux classifieurs d'apprentissage. Pour RF, nous voyons que OSBNR surpasse les autres méthodes de ré-échantillonnage pour toutes les distributions des ensembles d'apprentissage et de test. De même pour MLP, le meilleur score est obtenu par MLP combiné avec OSBNR pour toutes les distributions des ensembles d'apprentissage et de test.

D'après les résultats, nous avons bien observé l'impact de l'approche OSBNR proposée sur les performances de chaque classifieur en la comparant aux méthodes de ré-échantillonnage existantes. Nos résultats confirment que les données déséquilibrées sont un facteur critique dans l'évaluation des mesures de performance. Afin d'éviter ou minimiser les estimations biaisées des performances, nous recommandons AUC, pour mesurer la performance de détection [43, 44].

Performance des méthodes de ré-échantillonnage en termes de temps de traitement

Le traitement du temps pour toutes les méthodes comparées est présenté dans la Figure 5.21. Il montre le temps d'exécution moyen des différentes méthodes de ré-

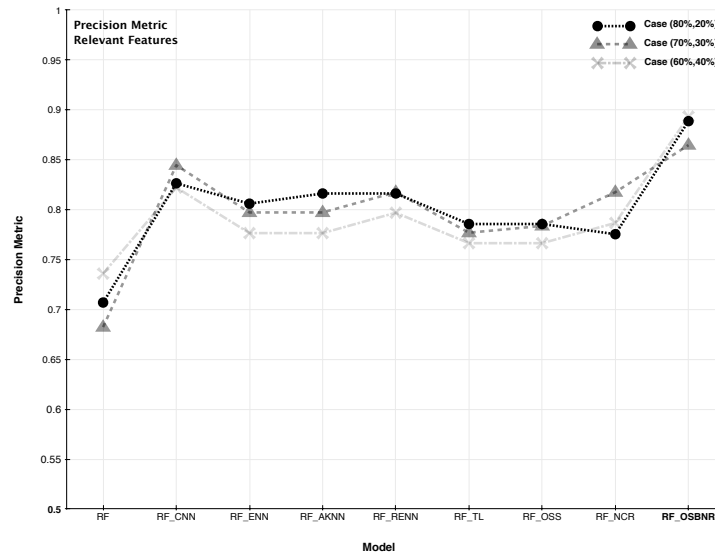


FIGURE 5.15 – Précision pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base, et cas des fonctionnalités pertinentes

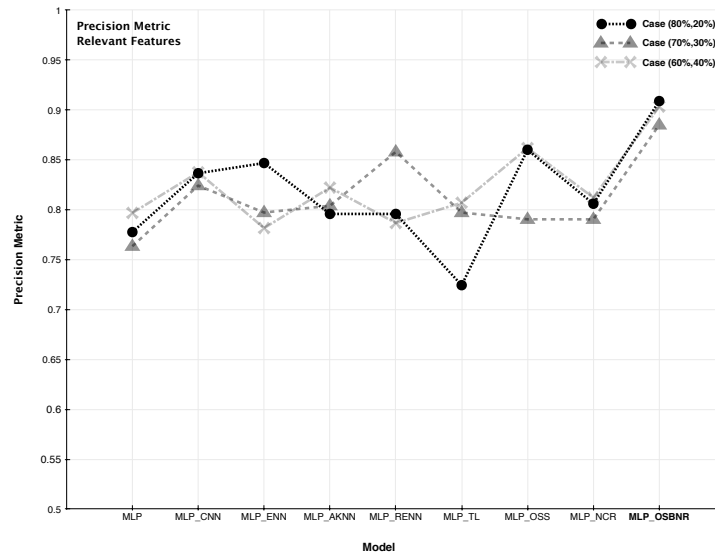


FIGURE 5.16 – Précision pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes

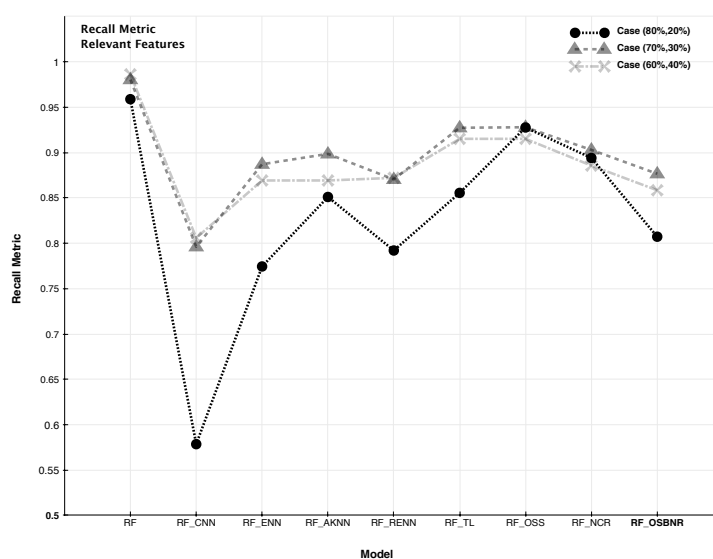


FIGURE 5.17 – Recall pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base, et cas des fonctionnalités pertinentes

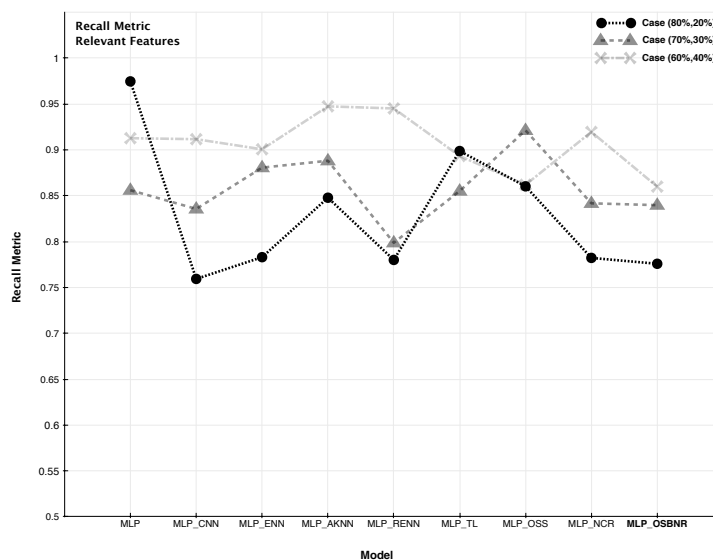


FIGURE 5.18 – Recall pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes

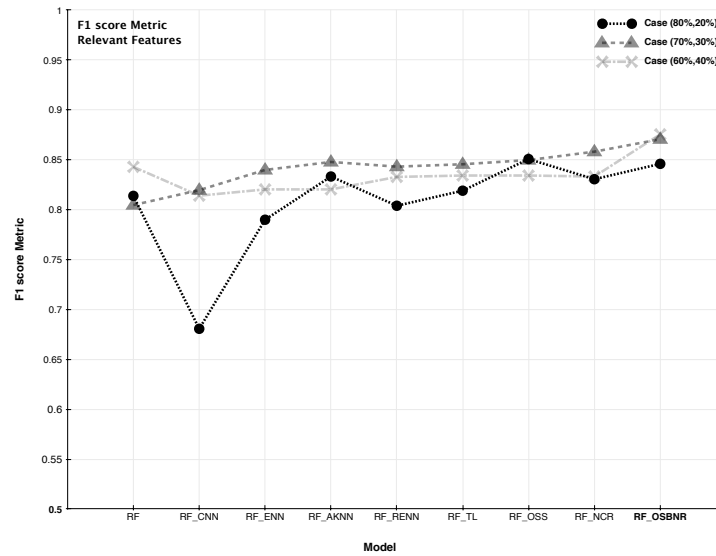


FIGURE 5.19 – F1-score pour OSBNR et les approches de rééchantillonnage de référence : RF comme classifieur de base, et cas des fonctionnalités pertinentes

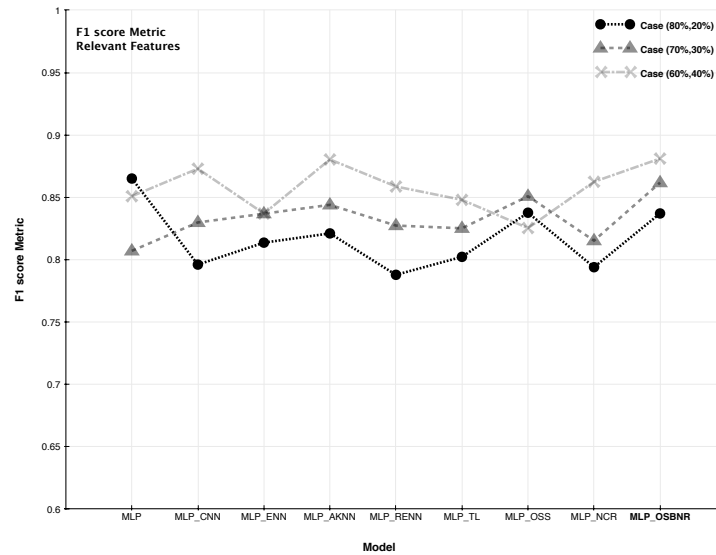


FIGURE 5.20 – F1-score pour OSBNR et les approches de rééchantillonnage de référence : MLP comme classifieur de base, et cas des fonctionnalités pertinentes

échantillonnage sur l'ensemble de données utilisé en tenant compte de toutes les fonctionnalités. D'après les résultats, nous pouvons voir que OSBNR est la technique la plus rapide parmi toutes les techniques de ré-échantillonnage comparées et OSS se classe deuxième. TL, ENN, NCL sont moyennement rapides pour le traitement. On remarque également que la méthode CNN est plus complexe car elle prend un certain temps de traitement important. En résumé, nous concluons qu'un classifieur combinée à notre approche OSBNR serait le choix optimal, offrant les meilleures performances de classification et nécessitant le moins de temps de traitement.

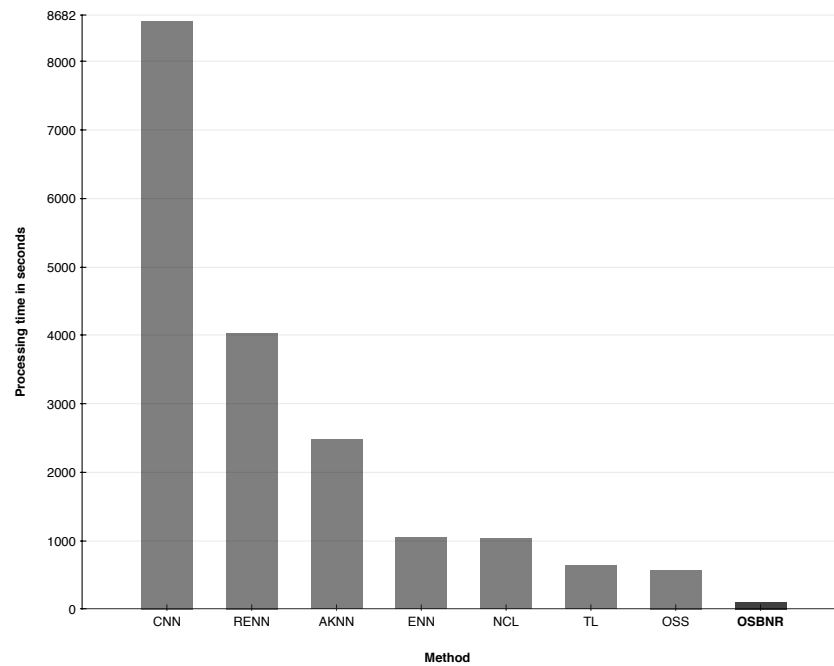


FIGURE 5.21 – Performance des méthodes de ré-échantillonnage : en terme du temps de traitement

5.4 Conclusion

Dans ce chapitre, nous avons abordé le problème de données déséquilibrés dans les systèmes de détection de fraude. L'apprentissage à partir de données déséquilibrées est l'un des problèmes le plus important à la fois du point de vue algorithmique et performance. Dans un tel contexte, la formation d'un classificateur conduit à négliger la classe minoritaire, qui est la classe d'intérêt lorsque l'on vise à faire face à la détection de fraude et d'anomalies. Un autre facteur critique qui accompagne le problème des données déséquilibrées est le chevauchement de classes et nous l'avons appelé : « bruit comportemental ». Le chevauchement de classes joue un rôle important dans la dégradation des performances lorsqu'il est associé à un déséquilibre de classe, ce qui rend très difficile voire impossible la distinction entre les deux classes.

Afin de remédier à ce problème, nous avons présenté une nouvelle approche appelée : One Side Behavioral Noise Reduction (OSBNR), de pré-traitement de données basée sur le clustering et le ré-échantillonnage afin de traiter le problème du déséquilibre et le chevauchement de classes. Notre approche est basée sur deux étapes. Premièrement, une analyse de clustering est appliquée pour grouper des instances similaires de la classe minoritaire dans plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de la classe minoritaire. Une série d'expériences a été réalisée pour valider la méthode proposée. Les résultats obtenus ont été comparés à plusieurs méthodes de ré-échantillonnage de référence qui nous a permis de mieux évaluer notre approche qui s'est révélée prometteuse comme solution au problème du déséquilibre de classe.

Dans le chapitre suivant, nous présentons une méthodologie de construction de modèles ML pour la détection de fraudes dans une approche de plate-forme développement basée sur des logiciels open-sources.

Chapitre 6

Système de détection de fraudes basé sur l'apprentissage automatique. Une approche méthodologique pour un système en production.

Sommaire

6.1	Introduction	112
6.2	Verrous liées aux données dans le système de detection d'anomalie	114
6.3	Méthodologie d'apprentissage automatique et application à la détection des fraudes	115
6.3.1	Collecte et stockage des données	115
6.3.2	Le pré-traitement des données	115
6.3.3	Construction du modèle	116
6.3.4	Évaluation du modèle	117
6.4	Credit card fraud detection use case : Experimentation and Evaluation	118
6.4.1	Collecte des données	118
6.4.2	Le prétraitement des données	118
6.4.3	Construction du modèle	119
6.4.4	Évaluation du modèle	119
6.4.5	Analyse des résultats	120
6.5	Plateforme de développement de modèles ML	121
6.5.1	Architecture générale	121
6.5.2	Système distribuée de données	122
6.5.3	Système distribuée de traitements	122
6.5.4	Algorithmes de Spark ML	123
6.5.5	Environnement de développement intégré	124
6.6	Conclusion	125

6.1 Introduction

Ces dernières années, le volume des transactions par cartes bancaires a considérablement augmenté en raison de la vulgarisation de leurs usages, du développement rapide des services associés, comme l'e-commerce, l'e-finance et tous les paiements mobiles. La Figure 6.1 (source [98]) présente la valeur totale des transactions par régions de croissance mondiale depuis 2017 et des estimations pour 2021-2022. Il existe une tendance nette à la hausse qui est nettement plus forte en Asie et dans la MEA. Dans ces régions, où le développement du réseau de cartes est relativement immature, la prolifération de l'utilisation des cartes est principalement due à l'augmentation des paiements mobiles et des portefeuilles en ligne [98].

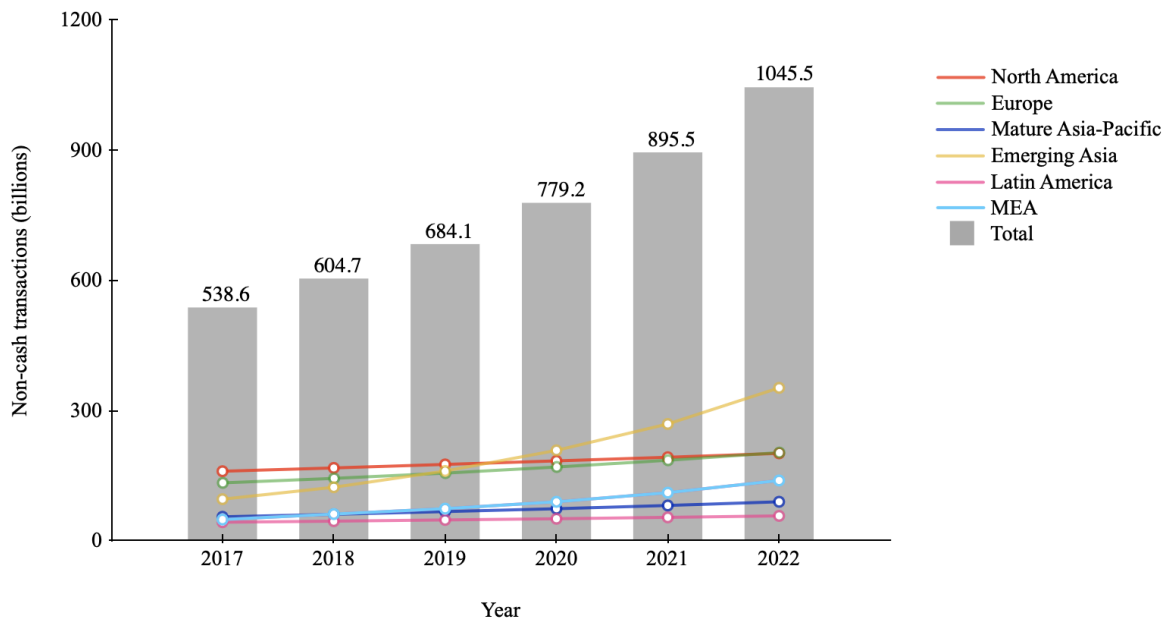


FIGURE 6.1 – Valeur des transactions non monétaire dans le monde de 2017 à 2022

L'utilisation à grande échelle des cartes bancaires conjuguée au développement de différents cas d'usage où les transactions passent sans vérification ni supervision rigides conduisent inévitablement à des pertes importantes. On estime qu'en 2020, la fraude par carte de crédit a atteint un total de 32 milliards de dollars [98]. Dans ce domaine, il devient nécessaire, voire crucial, de traiter immédiatement les données collectées pour détecter toute fraude potentielle. Cependant, les outils de détection traditionnels sont incapables de gérer les méga-données capturées. Par conséquent, ils ne peuvent pas détecter les anomalies et les menaces. Dès lors, il est devenu essentiel de surmonter les goulots d'étranglement des techniques existantes en utilisant une nouvelle génération d'approches issues de l'intelligence artificielle, comme celles basées sur l'apprentissage automatique.

Ces approches doivent collaborer étroitement avec les techniques de traitement de données massives, dites "Big Data". Elles doivent aussi fournir des réponses en quasiment temps réel.

En définitif, nous pensons que les modèles d'apprentissage automatique, appelés aussi modèles Machine Learning (ML), peuvent apporter une solution fiable au problème de détection d'anomalies et donc son application à la détection de fraude par carte bancaire. En fonction des besoins et du domaine d'application du ML, il existe plusieurs scénarios concernant la façon dont ces modèles peuvent être construits et appliqués. Les travaux récents dans le domaine du ML mettent en lumière deux sortes de plateformes importantes : plateforme de développement et plateforme de déploiement.

Les plateformes de développement réduisent considérablement le temps et le coût de création de modèles ML. Elles garantissent l'organisation de tous les niveaux de maturité de ces modèles. Elles peuvent être prêtes à l'emploi ou sur mesure, basées sur des logiciels open-sources ou commerciaux. Certaines plateformes de développement peuvent agir également en tant que plateformes de déploiement de modèles et, dans quelques cas, en tant que plateformes de gestion du cycle de vie des modèles, mais leur coeur est d'exécuter des pipelines de construction de modèles. La plateforme de déploiement apporte les modèles à l'environnement de production. Supposons qu'un modèle ML a été créé et évalué sur une plateforme de développement donnée, l'étape suivante consiste à utiliser ce modèle dans une application en production. Cette dernière peut être une simple application web, embarquée dans un dispositif IoT, ou alors un service ou micro-service faisant partie d'une architecture complexe. C'est le rôle de la plateforme de déploiement.

Les deux plateformes, développement et déploiement, sont liées par ce qu'on appelle le pipeline d'apprentissage automatique. En effet, le cycle de vie d'un modèle ne doit pas s'arrêter à l'étape de développement et évaluation. Il est mis en production puis retourner à l'étape de développement autant que possible pour être raffiné et amélioré.

Dans ce travail, nous présentons une méthodologie de construction de modèles ML dans une approche de plateforme développement basée sur des logiciels open-sources. Cette plateforme évolutive dédiée à un système de détection de fraudes apporte trois contributions importantes :

- un workflow de détection de fraude par carte de crédit. Il contient la plupart des idées de conception communes dans les derniers Fraud Detection Systems (FDS), ce qui facilite grandement l'intégration des algorithmes de détection dans le flux de travail ;
- un cadre de travail à quatre couches qui comprend une couche de stockage distribué, couche de prétraitement de données, couche d'apprentissage ou de construction du modèle, et enfin une couche de test locale ;
- une plateforme de développement dont le noyau centrale est basé sur Hadoop et Spark. Avec ces technologies, nous sommes capables de construire un système évolutif et fiable. Les résultats expérimentaux montrent que ce système a le potentiel d'atteindre une performance durable.

Ce chapitre est structuré en quatre sections. Tout d'abord, nous décrivons les défis liés aux données et au système pour concevoir une bonne solution de détection de fraude. Deuxièmement, nous présentons notre méthodologie de création de modèles ML pour la détection de fraude. La troisième section est consacrée à notre plateforme de développement basée sur des solutions open-source issues du monde du Big-Data.

6.2 Verrous liées aux données dans le système de détection d'anomalie

La compréhension et la détection en temps opportun des schémas de fraude sont fondamentales pour réduire le problème croissant de fraude par carte de crédit. La conception d'un FDS efficace, en temps réel et évolutif est soumise à plusieurs défis liés aux données et au système, ont été identifiés et énumérés dans cette section [85, 102, 1, 94, 42] :

- Déséquilibre des données : Comme mentionné précédemment, au niveau mondial, les transactions frauduleuses représentent moins de 0,1% du total des transactions [41]. Ce ratio est retenu dans la distribution de tout ensemble de données de fraude par carte de crédit, ce qui entraîne des classes très déséquilibrées. Si ce problème n'avait pas été pris en compte, tout algorithme d'apprentissage automatique qui ne classe correctement que les transactions authentiques effectuerait des performances exceptionnelles, avec un niveau de précision supérieur à 99%, sans tenir compte du fait que toutes les transactions de classe minoritaire sont faussement classées. Un bon système de détection des fraudes devrait être capable de gérer des distributions asymétriques des données. Ce défi peut être résolu en utilisant des méthodes de ré-échantillonnage des données pour réduire le déséquilibre ou garder l'ensemble d'apprentissage et utiliser une famille d'algorithmes d'apprentissage capable de traiter ce type de données fortement déséquilibré.
- Bruit de données : un ensemble de données est dit bruyant s'il contient des données corrompues ou erronées. Un bon système de détection des fraudes devrait être capable de gérer le bruit. En général, le bruit affecte les performances d'un classifieur.
- Chevauchement des données : En effet, certains schémas frauduleux ressemblent beaucoup aux schémas normaux [102]. Un bon système de détection des fraudes devrait être en mesure de détecter les transactions frauduleuses qui ressemblent beaucoup à des transactions légitimes.
- Distribution de non stationnarité : le rapport de déséquilibre et les fonctionnalités des fraudes changent avec le temps, de sorte qu'un modèle qui fonctionne bien pendant un intervalle de temps peut rapidement devenir obsolète et commencer à produire des prédictions inexactes. Par conséquent, le système de détection de fraude ne doit pas être statique, c'est-à-dire construit une fois et jamais mis à jour. Un bon système de détection des fraudes doit être dynamique ; il devrait pouvoir s'adapter aux changements des schémas frauduleux.
- Métrique d'évaluation : La mesure de classification utilisée pour évaluer les techniques de détection des fraudes doit être choisie avec soin. Ceci est dû au fait ;

des paramètres tels que la précision de la classification ne conviennent pas à une distribution asymétrique.

- Problème de coût de mauvaise classification : le coût de mauvaise classification de chaque transaction varie. La détection des fraudes est un domaine très sensible aux coûts. En général, les transactions frauduleuses non détectées sont beaucoup plus graves et coûteuses que la détection d'un comportement normal en tant que fraude. Par conséquent, un bon FDS doit prioriser les transactions avec un coût de classification erroné plus élevé.
- Manque de FDS en temps réel : la plupart des FDS existants rapportés dans la littérature travaillent sur des données d'archives qui peuvent être utilisées pour piloter les futures politiques de sécurité. La fraude doit être détectée et bloquée en temps réel pour éviter de futures fraudes.

6.3 Méthodologie d'apprentissage automatique et application à la détection des fraudes

La détection de fraude est un problème de Big Data. Notre méthodologie ML et application à la détection des fraudes prennent en compte toutes les propriétés des données intrinsèques à un système Big Data. Il comprend quatre couches principales : la collecte et le stockage des données, le traitement des données, la construction de modèles et l'évaluation avant la pré-production.

6.3.1 Collecte et stockage des données

Cette couche consiste à collecter les données concernant les transactions effectuées par les clients. Dans leurs états brutes, elles contiennent des informations classées confidentielles. La grande majorité des systèmes de paiement fournissent des interfaces pour accéder aux données des transactions. Les règles légales en matière de protection de données doivent être respectées.

La qualité des modèles ML se mesure, entre autres, par la qualité des données à partir desquelles ils sont construits. Par conséquent, savoir utiliser les bonnes pratiques de collecte de données sont essentielles pour développer des modèles hautement performants. Les données doivent être, le plus possible, sans erreurs et contenir des informations pertinentes pour la tâche à accomplir [99].

6.3.2 Le pré-traitement des données

Le traitement des données est une tâche importante et prend du temps en raison de son importance pour la performance globale. L'objectif du traitement des données est de créer ce que l'on appelle des données préparées. Le traitement des données comprend diverses opérations, comme le montre la Figure 6.2 et qui sont comme suit :

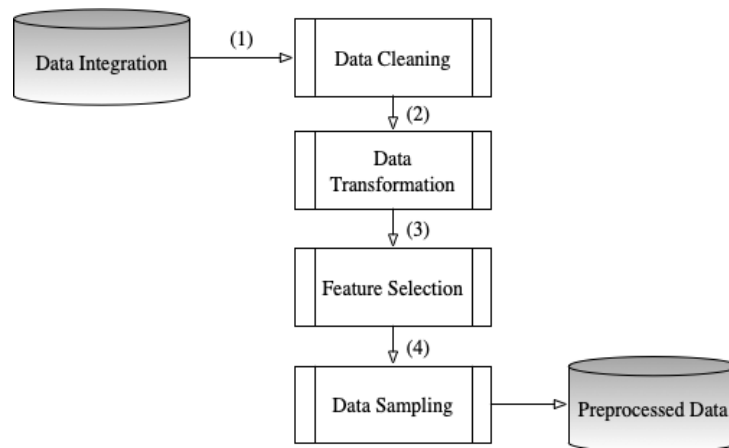


FIGURE 6.2 – Flux de prétraitement des données

1. Nettoyage des données : Suppression ou correction des enregistrements contenant des valeurs corrompues ou non valides dans les données brutes, et suppression des enregistrements pour lesquels il manque un grand nombre de colonnes.
2. Transformation des données : Conversion d'une fonctionnalité numérique en fonctionnalité catégorielle et conversion de fonctionnalités catégorielles en représentation numérique. Certains modèles fonctionnent uniquement avec des fonctionnalités numériques ou catégorielles, tandis que d'autres peuvent gérer des fonctionnalités de différents types.
3. Sélection des fonctionnalités : Sélection d'un sous-ensemble de fonctionnalités d'entrée pour l'entraînement du modèle tout en ignorant celles non pertinentes ou redondantes. Cela peut également impliquer la simple suppression de fonctionnalités si de nombreuses valeurs sont manquantes.
4. Échantillonnage des données : échantillonnage de l'ensemble de données avant d'entraîner le modèle prédictif afin d'avoir des données plus équilibrées. La dissymétrie et le chevauchement des données dans les cas de détection de fraudes sont les principales difficultés.

6.3.3 Construction du modèle

Le développement du modèle commence par partitionner les ensembles de données en un ensemble de données utilisé pour entraîner un modèle, un autre ensemble de données utilisé pour tester le modèle entraîné. Ce fractionnement des données garantit que le modèle ne mémorise pas un sous-ensemble particulier de données. Il y a deux étapes dans la construction d'un modèle d'apprentissage :

- Phase de formation : consiste principalement à construire ou à ajuster un modèle. Son objectif fondamental est d'apprendre un modèle ML les modèles récurrents à partir de données. L'algorithme d'apprentissage trouve des modèles dans les données

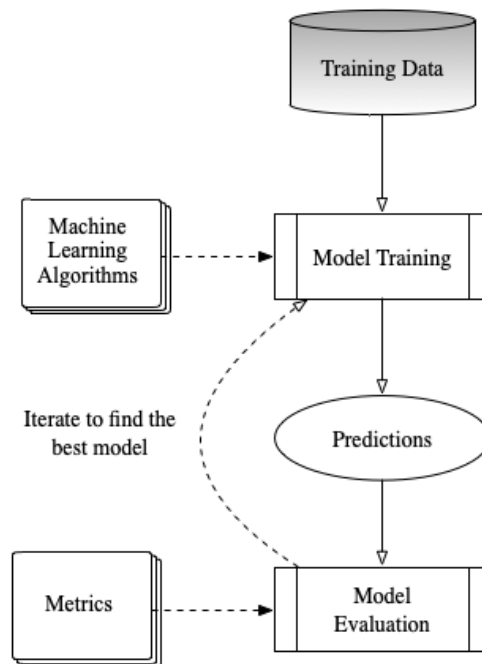


FIGURE 6.3 – Flux de phase de formation

d'apprentissage qui mappent les fonctionnalités de données d'entrée à la cible (la réponse de prédiction), et il génère un modèle ML qui capture ces modèles. (Fig. 6.3).

- Phase d'évaluation : consiste à évaluer le modèle formé concernant les observations de données invisibles. Une fois que le modèle est formé, il est important de vérifier s'il se comporte correctement sur des exemples inédits qui ne sont pas utilisés pour la formation du modèle. Pour ce faire, le modèle est utilisé pour prédire la réponse sur le jeu de données de test, puis la cible prédite est comparée à la réponse réelle pour juger la performance du modèle (Fig. 6.4).

6.3.4 Évaluation du modèle

Il est important de tester, mesurer et surveiller les performances d'un modèle prédictif. Il faut ensuite définir les mesures à utiliser pour l'évaluation de la performance. Les paramètres d'évaluation utilisés dépendent de plusieurs facteurs, tels que, la tâche de modélisation (classification, régression ou segmentation), le contexte du problème que nous essayons de résoudre ainsi que la distribution des données. Ces métriques sont également utilisées pour comparer les performances des modèles et sélectionner ceux qui offrent les meilleures performances.

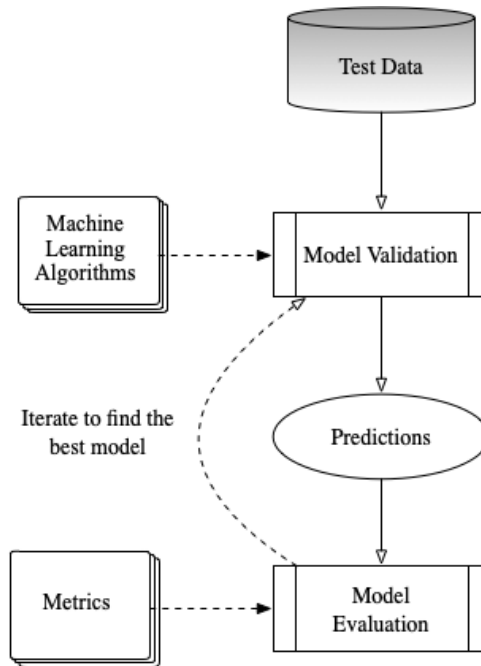


FIGURE 6.4 – Flux de phase d'évaluation

6.4 Credit card fraud detection use case : Experimentation and Evaluation

Dans cette section, nous appliquons notre méthodologie proposée, et discutons les résultats obtenus.

6.4.1 Collecte des données

Pour la collection des données, nous utilisons l'ensemble de données de détection de fraude par carte de crédit Kaggle [64] décrit dans la section 4.3.1 pour illustrer ces différentes étapes.

Après avoir collecté les données, nous chargerons ces données dans le cluster Hadoop. Ces données seront stockées de manière distribuée dans des nœuds de données. Une fois que les données sont chargées dans le cluster, elles sont prêtes pour un traitement ultérieur.

6.4.2 Le prétraitement des données

Au cours de la phase de prétraitement, après avoir examiné l'ensemble de données, nous avons procédé à supprimer les données manquantes. Puis, il a été démontré après analyse que toutes les fonctionnalités à l'exception de Amount et Time ont été mises à l'échelle à l'aide de la technique de transformation PCA. Par conséquent, les colonnes Time et Amount sont mis à l'échelle et normalisés pour assurer l'uniformité.

Pour la sélection de fonctionnalités, dans ce travail, nous ne nous intéresserons qu'aux fonctionnalités de données capables de séparer les deux classes (frauduleuse, normale). Les techniques de visualisation peuvent être utiles dans ce processus. Formellement, nous sélectionnons un sous-ensemble de fonctionnalités ou d'attributs dans l'ensemble de fonctionnalités et éliminons les fonctionnalités redondantes qui ne contribuent pas aux performances (section 4.3.3). Ainsi, une fonctionnalité est importante lorsque ses distributions de données des deux classes sont divergentes. Par conséquent, cette fonctionnalité peut potentiellement séparer les deux classes et améliorer les performances de prédiction.

La dernière étape de cette phase concerne l'échantillonnage de l'ensemble de données afin de réduire le déséquilibre de classe. La dissymétrie et le chevauchement des données dans les cas de détection de fraudes (i.e. le faible ratio d'actions frauduleuses sur l'ensemble de données) sont les principaux difficultés. Notre objectif est de classer correctement la classe minoritaire des transactions frauduleuses. Nous utilisons One Side Behavioral Noise Reduction (OSBNR) [44, 43] comme approche d'échantillonnage pour résoudre le problème du déséquilibre. elle permet de gérer le bruit comportemental afin d'améliorer la classification des instances de classe minoritaire. OSBNR consiste à séparer les transactions normales (instances de classe majoritaire) des transactions frauduleuses (instances de classe minoritaire). Ensuite, une analyse de cluster est appliquée pour regrouper les instances similaires de la classe minoritaire contenant les transactions frauduleuses dans plusieurs sous-ensembles, qui forment plusieurs groupes de comportement. La deuxième étape élimine les comportements de transaction normaux, considérés comme du bruit comportemental, qui se chevauchent avec les comportements de transactions frauduleuses.

6.4.3 Construction du modèle

Comme il n'y a pas de règle empirique pour diviser un ensemble de données en ensembles d'entraînement et de test, nous avons choisi la règle 70/30 pour les ensembles entraînement/test.

Plusieurs études ont rapporté Random Forest et Multilayer Perceptron pour obtenir les meilleures performances [41, 11, 29, 32], c'est l'une des raisons pour lesquelles nous adoptons Random Forest (RF) et Multilayer perceptron (MLP) dans notre expérience.

Les paramètres des modèles RF et MLP ont été déterminés à partir de divers tests préliminaires effectués sur les données d'entraînement, comme indiqué dans le Tableau 6.1.

6.4.4 Évaluation du modèle

Diverses métriques peuvent être utilisées pour mesurer la précision prédictive d'un modèle. Dans notre cas, le défi majeur est de s'attaquer au problème du déséquilibre, car les transactions légitimes sont beaucoup plus nombreuses que les transactions frauduleuses (moins de 1% du total des transactions). Ce problème conduit souvent à une exactitude extrêmement élevée où un modèle peut atteindre jusqu'à 99% d'exactitude. Afin d'éviter

TABLE 6.1 – Paramètres RF et MLP utilisés

Modèle ML	Paramètre
Random Forest (RF)	Nombre d'arbres = 20
	Profondeur de chaque arbre = 8
	Impureté = Gini
Multilayer Perceptron (MLP)	Nombre d'itérations = 100
	Paramètre de tolérance = 1e-6

ou minimiser les estimations biaisées des performances et d'après nos expériences précédentes, nous recommandons AUC pour mesurer la performance de détection. Elle s'est avérée efficace pour le déséquilibre de classe [67, 44, 43].

6.4.5 Analyse des résultats

Dans cette section, nous présentons les résultats générés pour évaluer l'évolutivité de notre méthodologie. En général, cette méthodologie obtient de bons résultats de classification ainsi qu'une bonne vitesse de traitement qui non seulement répond aux scénarios du monde réel, mais peut également être augmentée en ajoutant plus de machines au cluster. En plus d'être tolérant aux pannes et distribué. Les résultats sont présentés dans (Section 5.3) et résumés dans les sections suivantes.

6.4.5.1 Temps d'apprentissage et de prédiction

Les deux algorithmes prennent moins de temps pour l'apprentissage et la prédiction. à partir de la Figure 6.5, nous pouvons voir que le modèle de la forêt aléatoire donne le meilleur temps d'exécution total.

6.4.5.2 Qualité d'apprentissage

Les Figures 5.13 et 5.14 montrent les résultats obtenues par les modèles MLP et RF en utilisant la métrique AUC. Ils montrent également la performance du modèle avec l'approche OSBNR avec différentes méthodes de ré-échantillonnage pour étudier les performances des modèles ML résultants [43, 44].

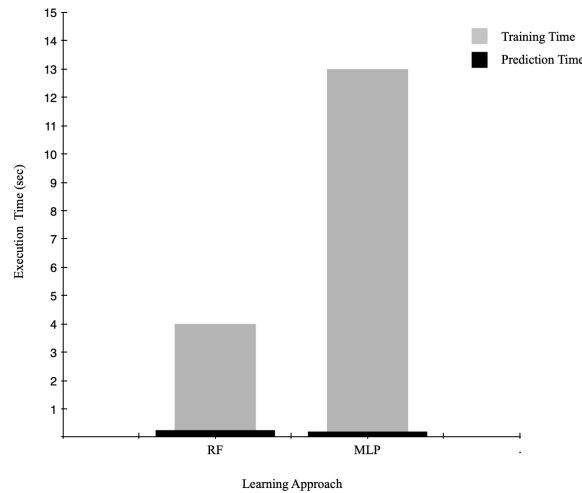


FIGURE 6.5 – Temps d'exécution total

6.5 Plateforme de développement de modèles ML

6.5.1 Architecture générale

La mise à l'échelle horizontale garantit la rentabilité de la plateforme lorsque les exigences en matière de données et de traitement varient en fonction des modèles. Ajouter plus de machines à un cluster n'aurait pas beaucoup de valeur en soi. Ce dont nous avons besoin, c'est un système capable de tirer parti de l'évolutivité horizontale et qui fonctionne sur plusieurs machines de manière transparente, indépendamment du nombre de machines dans le cluster. Le choix d'un système distribué pour la plateforme de développement de modèles ML s'impose alors car il fonctionne de manière transparente sur un cluster de machines et gère automatiquement les ressources nécessaires. Comme présenté précédemment dans la Figure 4.5, notre cluster est formé par un nœud maître et trois nœuds esclaves configurés comme suit :

- Système d'exploitation : Ubuntu-16.04.3-server-amd64
- Disque dur : type SSD 500 Go
- Processeurs : 16 Go de RAM (4 sockets, 4 cœurs)
- Réseau : 2x10 Go SFP +

Le cluster est géré par une plateforme de type cloud privé au sein du laboratoire L3i dans le cadre du projet FEDER-PLAIBDE [91]. Fondamentalement, le cluster est basé sur :

1. un système distribué de données ;
2. un système distribué de traitements ;
3. une bibliothèque d'algorithmes ML ;
4. un environnement de développement intégré.

6.5.2 Système distribuée de données

Dans cette conception, nous avons fait le choix du stockage de données en format brute. Ainsi, le système ne dépend d'aucun modèle de données. C'est donc tout naturellement que le choix de Apache Hadoop Distributed File System (HDFS) s'impose pour gérer notre système de données distribué.

Le HDFS est un exemple bien connu de système de fichiers distribué dans l'écosystème Big Data. Dans le HDFS, on utilise une architecture « maître / esclave » constituée d'un seul NameNode qui gère le système de fichiers distribué et de plusieurs DataNodes, qui résident généralement sur chaque nœud du cluster et gèrent les disques physiques attachés à ce nœud ainsi que les données qui sont conservées physiquement. Tout comme avec les systèmes de fichiers traditionnels, HDFS prend en charge les opérations standard du système de fichiers, telles que l'ouverture et la fermeture de fichiers et de répertoires. Lorsqu'une application cliente demande qu'un fichier soit écrit sur le HDFS, il est divisé en un ou plusieurs blocs qui sont ensuite mappés par le NameNode aux DataNodes, où ils sont physiquement persistants. Lorsqu'une application cliente demande qu'un fichier soit lu à partir du HDFS, les DataNodes répondent à cette demande.

Un des principaux avantages de HDFS est sa tolérance aux pannes liée à son architecture distribuée, ainsi qu'à la réplication des données. Comme il contient plusieurs nœuds, le cluster HDFS résiste aux pannes matérielles car les opérations peuvent être automatiquement déchargées vers les parties saines du cluster pendant que le matériel non fonctionnel est en cours de récupération ou de remplacement. En outre, lorsqu'un fichier est divisé en blocs et cartographié par le NameNode vers les DataNodes, ces blocs peuvent être configurés pour se répliquer automatiquement sur ces derniers, en tenant compte de la topologie du cluster HDFS.

6.5.3 Système distribuée de traitements

Dans une approche de traitement traditionnelle, les données doivent être déplacées vers la machine exécutant le processus. Le problème majeur de cette approche est que les données doivent être déplacées vers le code à chaque fois qu'un processus est exécuté. Par conséquent, l'approche traditionnelle de traitement ne convient pas aux problèmes nécessitant des volumes et de fréquences de données accrus. Nous avons donc besoin d'une autre approche de traitement de ces données. Une nouvelle approche où le processus est déplacé vers les données et qui fonctionne sur un cluster distribué. Autrement dit, une approche de traitement distribué.

L'idée fondamentale du traitement distribué est le concept de fractionnement d'une tâche de traitement en sous-tâches plus petites. Ces sous-tâches sont réparties dans le cluster et traitent des parties spécifiques des données. En règle générale, les sous-tâches sont situées sur les mêmes nœuds que les données elles-mêmes pour augmenter les performances. Les résultats des sous-tâches sont agrégés avant que le résultat final ne soit renvoyé.

Dans notre plateforme de développement de modèles ML, nous utilisons Apache Spark. C'est moteur de traitement distribué polyvalent capable de gérer des volumes de données importants. Le côté polyvalent de Apache Spark réside dans le fait qu'il convient à une grande variété de cas d'utilisation à grande échelle. Dans ce travail, nous nous intéressons particulièrement à sa bibliothèque d'apprentissage automatique appelée « Spark ML ».

Spark utilise une architecture « maître / esclave » avec un coordinateur central appelé « Driver » et un ensemble de workflows exécutables appelés « Executors » situés sur différents nœuds du cluster. Chaque application écrite dans Apache Spark donne lieu à un programme pilote ou « Driver ». Il segmente une application Spark en tâches qui sont ensuite partitionnées entre les nœuds Worker dans le cluster distribué. Le programme pilote crée également un SparkContext qui indique à l'application comment se connecter au cluster et à ses services sous-jacents. Les nœuds esclaves sont l'endroit où le traitement de calcul se produit physiquement. En règle générale, ils sont situés sur les mêmes nœuds que ceux où les données sous-jacentes sont disponibles. Ces nœuds génèrent des processus appelés exécuteurs « Spark Executor ». Ils sont responsables de l'exécution des tâches de calcul et du stockage des données mises en cache localement. Les exécuteurs communiquent avec le programme pilote afin de recevoir des fonctions programmées qui sont ensuite exécutées.

Le gestionnaire de cluster est un processus qui contrôle, régit et réserve les ressources sous la forme de conteneurs sur le cluster. Ces conteneurs sont réservés à la demande du maître d'application « Application Master ». Une fois les conteneurs alloués par le gestionnaire, le maître d'application transfère les ressources du conteneur au pilote Spark. Apache Spark est fourni avec son propre gestionnaire de cluster simple qui, lorsqu'il est utilisé, est appelé mode Spark Standalone. Spark peut se connecter à différents types de gestionnaires de cluster, tels que Mesos ou Kubernetes. Dans notre architecture, nous utilisons le gestionnaire fourni par Apache Hadoop appelé YARN.

6.5.4 Algorithmes de Spark ML

Spark ML est une brique logicielle incontournable de notre plateforme de développement de modèles ML. Elle offre tous les services dédiés à la mise en oeuvre de ces modèles. Dans Apache Spark, peut distinguer deux API :

1. Spark ML : basée sur l'API DataFrame. Il s'agit d'une structure de données distribuées, orientée colonne et adaptée pour les algorithmes d'apprentissage.
2. Spark MLlib : basée sur l'API RDD (Resilient Distributed Datasets). Il s'agit d'une de données distribuées, orientée objet représentant les données.

Dans le cadre de notre plateforme de développement de modèles ML, nous utilisons Spark ML. Au coeur de celle-ci, on trouve plusieurs couches logicielles pour différents services. Parmi celles-ci, on peut en citer trois parties fondamentales :

- Transformers : ce sont des algorithmes permettant de transformer une DataFrame en une autre.
- Estimators : formellement, un estimateur fait abstraction d'un algorithme d'apprentissage. En pratique, un estimateur est un type d'algorithme d'apprentissage.

- Evaluators : ce sont des mesures permettant d'évaluer la performance d'un modèle d'apprentissage.

6.5.5 Environnement de développement intégré

Dans notre plateforme de développement de modèles ML, nous avons fait le choix d'utiliser un environnement de développement basé sur Jupyter Lab. C'est une application web permettant d'enregistrer tout le processus de fabrication, de test, d'évaluation et documentation d'un modèle. Le produit Jupyter a été développé à l'origine dans le cadre du projet IPython. Il permet le développement interactif en plusieurs langages. Le nom Jupyter lui-même est dérivé de la combinaison de Julia, Python et R.

Le cluster Spark de notre plateforme de développement de modèles ML fournit trois noyaux utilisés avec Jupyter Lab sur Apache Spark. Un noyau est un programme qui exécute et interprète le code source d'un modèle. Les trois noyaux sont :

- PySpark : pour les applications écrites en Python2.
- PySpark3 : pour les applications écrites en Python3.
- Spark : pour les applications écrites en Scala.

Le noeud indépendant, fournissant le service Jupyter dans notre plateforme, utilise l'API REST Apache Spark pour soumettre des tâches distantes à un notre cluster Spark. Pour cela, on utilise Apache Livy. C'est un service qui permet une interaction facile avec un cluster Spark via une interface REST. Il permet de soumettre facilement des travaux Spark ou des extraits de code Spark, la récupération de résultats synchrone ou asynchrone, ainsi que la gestion du contexte Spark, le tout via une simple interface REST.

Pour communiquer à distance avec le serveur Apache Livy, le service Jupyter dans notre plateforme utilise un programme client appelé Sparkmagic. Lorsque nous écrivons du code Spark sur notre client Jupyter local, Sparkmagic exécute le travail Spark via Livy. On peut donc utiliser Spark à partir de tout environnement Jupyter. Nous n'avons besoin d'aucune configuration Spark provenant du cluster. Nous pouvons donc exécuter une tâche Spark dans un cluster comme une exécution sur une machine locale.

6.6 Conclusion

Les activités bancaires frauduleuses sont le plus gros casse-tête dans le monde actuel en raison des progrès technologiques. Le mécanisme de détection de la fraude avec une plus grande précision et une technique de prévision rapide est essentiel dans le secteur bancaire pour attirer plus de clients, promettre leur sécurité et améliorer leur statut auprès du public et des autres institutions financières.

Dans ce chapitre nous avons présenté une méthodologie de conception d'un système de détection de fraude (FDS) efficace basé sur les technologies Big Data. Nous avons abordé le problème de détection des fraudes et sa résolution en utilisant une approche ML. De plus, nous avons montré que cette approche nécessite la conception et la mise en œuvre d'une plateforme adéquate. Nous avons proposé une méthodologie générale basée sur la formalisation, la mise en œuvre et la conception technique d'une plateforme de détection de fraude. La partie formalisation se compose de quatre couches : le stockage distribué, le traitement des données, la construction du modèle et enfin l'évaluation du modèle. La partie implémentation utilise le système de traitement de données distribué Spark. En particulier, nous nous sommes basé sur son cadre dédié au Machine Learning, appelé MLlib. La partie conception technique de la plateforme s'appuie sur les dernières technologies Big Data telles que Hadoop, Yarn, Livy. La solution présentée est basée sur des solutions open-source issues du monde du Big-Data.

Dans le chapitre suivant, nous faisons une synthèse des apports de cette thèse, nous récapitulons les problématiques de notre recherche et comment nos travaux permettent d'y répondre et nous explicitons les contributions apportées durant ces années. Nous abordons ensuite les perspectives d'évolutions des travaux futurs sur le sujet.

Conclusion

Dans cette partie, nous avons introduit nos contributions. Dans le chapitre 4, nous avons réalisé une évaluation expérimentale de performances des techniques d'apprentissage automatique pour identifier leurs faiblesses lorsqu'ils sont confrontés au problème du déséquilibre de classes. Nous avons montré l'impact du déséquilibre des données sur la performance des modèles ML. Deuxièmement au chapitre 5, nous avons introduit une nouvelle approche appelée : OSBNR (One Side Behavioral Noise Reduction), afin de traiter le problème du déséquilibre de classe. Notre méthode OSBNR (One Side Behavioral Noise Reduction) est établie en deux étapes. Premièrement, une analyse de regroupement ou clustering est appliquée pour regrouper des instances similaires de la classe minoritaire dans plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de la classe minoritaire.

Enfin dans le chapitre 6, nous avons présenté une méthodologie de construction de modèles ML pour la détection de fraudes dans une plateforme développement basée sur des logiciels open-sources.

Conclusion et perspectives

Conclusion

La carte de crédit est l'un des modes de paiement populaires pour les transactions électroniques dans de nombreux pays développés et en développement. L'invention des cartes de crédit a rendu les transactions en ligne fluides, plus faciles, confortables et pratiques. Cependant, il a également offert de nouvelles possibilités de fraude aux criminels et, par conséquent, a augmenté le taux de fraude. L'impact mondial de la fraude par carte de crédit est alarmant, des millions de dollars ont été perdus par de nombreuses entreprises et particuliers.

Les progrès récents dans les approches d'intelligence artificielle, tels que l'apprentissage automatique, font une percée importante dans ce domaine. L'apprentissage automatique a joué un rôle impératif dans la détection d'anomalies ou fraude, offrant une pléthore de techniques pour trouver des modèles dans les données, distinguant les transactions normales des transactions suspectes.

La détection de fraude par carte de crédit, qui est un problème de la science de données et d'apprentissage automatique, est une tâche particulièrement difficile et complexe du point de vue de l'apprentissage pour plusieurs raisons principales. L'un des principaux défis de la détection de fraude consiste à gérer de manière appropriée la distribution asymétrique des données. Autrement dit, il existe de nombreuses transactions légitimes et peu de preuves de transactions frauduleuses dont on peut tirer des leçons, ce qui complique le processus de détection. De nombreuses études de détection de fraude par carte de crédit font état d'un taux de fraude inférieur à 0,5%.

De plus, Les transactions frauduleuses présentent souvent les mêmes caractéristiques que les transactions légitimes, qui représente un autre problème critique qui accompagne le déséquilibre de classes. ont formulé cela comme la présence de données qui se chevauchent et nous l'avons appelé : « bruit comportemental ». Notamment, soit parce que les fraudeurs gardent le comportement normal pour passer inaperçu et rester couverts par des non-fraudeurs, ou les changements de mode de vie des clients peuvent affecter les habitudes de dépenses. Par conséquent, les modèles qui comparent une nouvelle transaction à l'historique des transactions du client marquent les changements dans les habitudes de dépenses comme suspects. Cela conduit à une classification erronée qui peut être coûteuse.

Dans cette thèse, nous avons abordé le problème de l'apprentissage à partir de données déséquilibrées en présence d'un niveau de bruit comportemental dans le contexte de la détection des fraudes. Le travail réalisé au cours de cette thèse se voulait éclectique et explorait différentes façons d'aborder le problème de l'apprentissage dans ce contexte non trivial. Nos contributions sont principalement réparties en deux catégories :

Contributions principales

Des contributions principales peuvent être résumées comme suit :

- OSBNR : nous avons présenté, dans le chapitre 5, une nouvelle approche appelée : One Side Behavioral Noise Reduction (OSBNR), de pré-traitement de données basée sur le clustering et le ré-échantillonnage afin de traiter le problème du déséquilibre et le chevauchement de classes. Notre approche est basée sur deux étapes. Premièrement, une analyse de clustering est appliquée pour grouper des instances similaires de la classe minoritaire dans plusieurs clusters de comportement. Deuxièmement, nous sélectionnons et éliminons les instances de la classe majoritaire, considérées comme du bruit comportemental, qui se chevauchent avec les clusters de la classe minoritaire. Une série d'expériences a été réalisée pour valider la méthode proposée. Les résultats obtenus ont été comparés à plusieurs méthodes de ré-échantillonnage de pointe qui nous a permis de mieux évaluer notre approche qui s'est révélée prometteuse comme solution au problème du déséquilibre de classe. (Voir pour plus de détails.)
- Projet open-source spark-OSBNR : implémentation OSBNR en scala sous Spark, mis à la disposition de la communauté de recherche. Plus d'informations sur ce projet sont présentées à l'annexe B. Il s'agit d'une contribution utile, car les packages Spark pour travailler avec des données déséquilibrées sont rares.
- Conception FDS : cette contribution couvre l'implémentation technique d'un système de détection de fraude utilisant des technologies open-sources Hadoop et Spark introduit dans le chapitre 6. La conception, la mise en œuvre sont inclus ici. Notre implémentation dans Spark est hautement évolutive et fiable, ce qui est essentiel pour gérer la grande taille des ensembles de données du monde réel. Le processus d'obtention du meilleur modèle à mettre en œuvre a été décrit, Il contient la plupart des idées de conception communes dans les derniers Fraud Detection Systems (FDS), ce qui facilite grandement l'intégration des algorithmes de détection dans le flux de travail. Les résultats expérimentaux montrent que ce système a le potentiel d'atteindre une performance durable.

Contributions secondaires

En plus des principales contributions discutées ci-dessus, nous avons mené une étude expérimentale et une étude de la littérature concernant les techniques ML pour détection de fraude et l'apprentissage dans un contexte de données déséquilibrées. Celles-ci sont essentiellement fortement liées à la contribution principale. Les études sont brièvement expliquées ci-dessous :

- Étude bibliographique : Nous avons mené une étude rigoureuse de la littérature pour identifier la force et les faiblesses des techniques ML existantes. L'étude a couvert les approches ML qui ont été adoptées pour la détection de la fraude par carte de crédit. nous avons discuté les deux grandes catégories d'algorithmes d'apprentissage avec leurs avantages et inconvénients. Deuxièmement, nous avons cité les différentes méthodes techniques proposées dans la littérature pour traiter le problème de données déséquilibrées, qui sont regroupées en deux grandes familles : les méthodes opérant

au niveau de données et les méthodes opérant au niveau algorithmique. Enfin, nous avons introduit plusieurs métriques de performance bien adaptées dans ce contexte.

- Étude expérimentale : Nous avons évalué de manière approfondie les modèles d'apprentissage supervisé et non supervisé pour la détection d'anomalies dans les transactions bancaires par carte de crédit en particulier. Nous avons mené une étude expérimentale en utilisant diverses approches supervisées et non supervisées. Plus précisément, 7 modèles de classification supervisés, à savoir, régression logistique (LR), Support Vector Machines (SVM), Decision Tree (DT), Random Forest (RF), Gradient-Boosted Tree (GBT), Naïve Bayes (NB) et Multilayer Perceptron (MLP) ainsi que 3 modèles de détection d'anomalies non supervisés, à savoir, K-Means (KM), Bisecting K-Means (BKM), et Gaussian Mixture Model (GMM), sont explorés dans cette étude. Nous avons évalué leurs performances sur un ensemble de données de transaction par carte de crédit Kaggle⁸ en tenant compte de la distribution des anomalies dans un jeu de données déséquilibré. Cette étude expérimentale a été très efficace pour comprendre les véritables faiblesses des solutions existantes concernant le problème de déséquilibre de classe. Nous avons montrés l'impact du déséquilibre des données sur la performance de ces approches. Ainsi, à partir des résultats obtenus, on note l'existence d'un ensemble de paramètres adaptés aux modèles supervisés et d'autres aux modèles non supervisés. En effet, nos expériences ont mis en évidence que Random Forest et MultiLayer perceptron sont les méthodes les plus prometteuses.

8. Plateforme la plus populaire pour les compétitions de science des données

Perspectives

Avec le travail réalisé par cette thèse, nous avons réussi à résoudre certains des problèmes les plus difficiles auxquels sont confrontés les techniques d'apprentissage pour la détection de fraude. Nous souhaitons apporter quelques extensions possibles des apports de cette thèse, qui pourraient élargir les connaissances et résoudre certaines des limites techniques de ce travail.

- Création d'un ensemble de données collaboratif de détection de fraude : L'absence de grands ensembles de données réels dans le domaine de la fraude pour la communauté de recherche entrave la recherche de nouvelles approches pratiques de détection. Il est suggéré qu'il devrait y avoir un objectif de faciliter la coopération entre les chercheurs et le monde commercial pour rendre ces ensembles de données accessibles au public avec autorisation, lorsque ceux-ci ont été suffisamment obscurcis pour surmonter les problèmes de sécurité et de protection des données. Ces ensembles de données pourraient être utilisés comme standard pour comparer les résultats académiques dans la tâche de détection de fraude.
- Comparer les différents cadres distribués : cela pourrait aider à surmonter les limitations des algorithmes implémentés dans Spark ML, utiliser d'autres bibliothèques spécialisées uniquement sur ML pourrait être mieux. Par exemple, H2O implémente des algorithmes ML distribués et possède plus de fonctionnalités que Spark ML. Comparer ces algorithmes et étudier leurs différences pourraient être intéressants. Il est à noter que H2O peut être facilement intégré à Spark en utilisant Sparkling Water, ce qui facilite la comparaison entre eux car la préparation des données peut être partagée.
- Explorer d'autres approches de détection de fraude : cette approche aborde le problème de détection de fraude sous l'angle d'un problème de classification binaire. Cependant, d'autres approches peuvent également être utilisées et pourraient être préférées dans d'autres cas d'utilisation. Les analyser en profondeur et les comparer pourrait être très intéressant. Certaines des suggestions pour une étude plus approfondie sont :
 1. Analyse de graphe : Récemment, les techniques d'apprentissage automatique ont été étendues aux structures de données de types graphes. L'ensemble des transactions est lui-même un grand graphe attribué parmi lesquelles nous cherchons à repérer des transactions frauduleuses qui sont considérées comme des anomalies (outliers).
 2. Analyse semi-supervisée : utiliser l'apprentissage supervisé et non supervisé pour améliorer la performance et la détection de nouvelles stratégies de fraude.
 3. Analyse en profondeur : L'apprentissage en profondeur a récemment fait l'objet de nombreuses recherches, en particulier dans des applications telles que la reconnaissance d'images et le traitement du langage naturel. Cependant, il n'est pas clair si cette méthode présente des avantages dans le domaine de la détection des fraudes. La nature temporelle et séquentielle des transactions

est importante car les humains développent des comportements habituels, où des modèles de dépenses pour certains biens, magasins, marques, montants peuvent être observés sur une période. Comme le système de détection de fraude fonctionne généralement en temps réel sur un flux de données, il s'agit d'un domaine clé d'amélioration.

- Recherche sur les métriques de détection des fraudes : il n'existe pas de métrique d'évaluation standard pour la détection des fraudes. Il serait intéressant d'effectuer une comparaison formelle entre les métriques basées sur les coûts et les métriques traditionnelles.
- Généralisation de notre méthode OSBNR sur d'autres domaines d'application.

Je voudrais terminer cette thèse par une citation d'Eliezer Yudkowsky :

“Anything that could give rise to smarter-than-human intelligence - in the form of Artificial Intelligence, brain-computer interfaces, or neuroscience-based human intelligence enhancement - wins hands down beyond contest as doing the most to change the world. Nothing else is even in the same league.”

Available Code

<https://github.com/salmaelhajjami/>

Bibliographie

- [1] Adewumi, A. O. & Akinyelu, A. A. (2017). A survey of machine-learning and nature-inspired based credit card fraud detection techniques. *International Journal of System Assurance Engineering and Management*, 8(2), 937–953.
- [2] Agaskar, V., Babariya, M., Chandran, S., & Giri, N. (2017). Unsupervised learning for credit card fraud detection. *International Research Journal of Engineering and Technology (IRJET)*, 4, 2343–2346.
- [3] Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60, 19–31.
- [4] Ahmed, M., Mahmood, A. N., & Islam, M. R. (2016). A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*, 55, 278–288.
- [5] Azencott, C.-A. (2019). *Introduction au machine learning* : Dunod.
- [6] Baesens, B., Van Vlasselaer, V., & Verbeke, W. (2015). *Fraud analytics using descriptive, predictive, and social network techniques : a guide to data science for fraud detection* : John Wiley & Sons.
- [7] Bangoria, B., Mankad, N., & Pambhar, V. (2014). Enhanced k-means clustering algorithm to reduce time complexity for numeric values. *International Journal of Computer Science and Information Technologies*, 5(1), 876–879.
- [8] Bansal, M. (2014). Credit card fraud detection using self organised map. *International Journal of Information & Computation Technology*, 4(13), 1343–1348.
- [9] Behera, T. K. & Panigrahi, S. (2015). Credit card fraud detection : a hybrid approach using fuzzy clustering & neural network. In *2015 Second International Conference on Advances in Computing and Communication Engineering*, (pp. 494–499). IEEE.
- [10] Bekkar, M., Djemaa, H. K., & Alitouche, T. A. (2013). Evaluation measures for models assessment over imbalanced data sets. *J Inf Eng Appl*, 3(10).
- [11] Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud : A comparative study. *Decision Support Systems*, 50(3), 602–613.
- [12] Błaszczyński, J. & Stefanowski, J. (2015). Neighbourhood sampling in bagging for imbalanced data. *Neurocomputing*, 150, 529–542.

- [13] Breiman, L. (2001). Statistical modeling : the two cultures. *Statist. Sci.*, 16(3), 199–231.
- [14] Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees* : CRC press.
- [15] Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2009). Safe-level-smote : Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Pacific-Asia conference on knowledge discovery and data mining*, (pp. 475–482). Springer.
- [16] Cabanes, G., Bennani, Y., & Grozavu, N. (2013). Unsupervised learning for analyzing the dynamic behavior of online banking fraud. In *2013 IEEE 13th International Conference on Data Mining Workshops*, (pp. 513–520). IEEE.
- [17] Cao, P., Zhao, D., & Zaiane, O. (2013). An optimized cost-sensitive svm for imbalanced data learning. In *Pacific-Asia conference on knowledge discovery and data mining*, (pp. 280–292). Springer.
- [18] Carcillo, F., Le Borgne, Y.-A., Caelen, O., Kessaci, Y., Oblé, F., & Bontempi, G. (2019). Combining unsupervised and supervised learning in credit card fraud detection. *Information Sciences*.
- [19] Carneiro, N., Figueira, G., & Costa, M. (2017). A data mining based system for credit-card fraud detection in e-tail. *Decision Support Systems*, 95, 91–101.
- [20] Castro, C. L. & Braga, A. P. (2013). Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data. *IEEE transactions on neural networks and learning systems*, 24(6), 888–899.
- [21] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection : A survey. *ACM computing surveys (CSUR)*, 41(3), 1–58.
- [22] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote : synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16, 321–357.
- [23] Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Special issue on learning from imbalanced data sets. *ACM SIGKDD explorations newsletter*, 6(1), 1–6.
- [24] Chawla, N. V., Lazarevic, A., Hall, L. O., & Bowyer, K. W. (2003). Smoteboost : Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery*, (pp. 107–119). Springer.
- [25] Chiang, M.-C., Tsai, C.-W., & Yang, C.-S. (2011). A time-efficient pattern reduction algorithm for k-means clustering. *Information Sciences*, 181(4), 716–731.
- [26] Chuang, P.-J. & Wu, D.-Y. (2019). Applying deep learning to balancing network intrusion detection datasets. In *2019 IEEE 11th International Conference on Advanced Infocomm Technology (ICAIT)*, (pp. 213–217). IEEE.

- [27] Cleveland, W. S. (2001). Data science : an action plan for expanding the technical areas of the field of statistics. *International Statistical Review*, 69(1), 21–26.
- [28] credit card processing, S. Credit card fraud statistics.
- [29] Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2015). Credit card fraud detection and concept-drift adaptation with delayed supervised information. In *International joint conference on Neural networks*, (pp. 1–8). IEEE.
- [30] Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2017). Credit card fraud detection : a realistic modeling and a novel learning strategy. *IEEE transactions on neural networks and learning systems*, 29(8), 3784–3797.
- [31] Dal Pozzolo, A., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. In *2015 IEEE Symposium Series on Computational Intelligence*, (pp. 159–166). IEEE.
- [32] Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S., & Bontempi, G. (2014). Learned lessons in credit card fraud detection from a practitioner perspective. *Expert systems with applications*, 41(10), 4915–4928.
- [33] Denil, M. & Trappenberg, T. (2010). Overlap versus imbalance. In *Canadian conference on artificial intelligence*, (pp. 220–231). Springer.
- [34] Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 54, 764–771.
- [35] Dhankhad, S., Mohammed, E., & Far, B. (2018). Supervised machine learning algorithms for credit card fraudulent transaction detection : a comparative study. In *International Conference on Information Reuse and Integration (IRI)*, (pp. 122–125). IEEE.
- [36] Dheepa, V. & Dhanapal, R. (2009). Analysis of credit card fraud detection methods. *International journal of recent trends in engineering*, 2(3), 126.
- [37] Domingos, P. (1999). Metacost : A general method for making classifiers cost-sensitive. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, (pp. 155–164).
- [38] Douzas, G., Bacao, F., & Last, F. (2018). Improving imbalanced learning through a heuristic oversampling method based on k-means and smote. *Information Sciences*, 465, 1–20.
- [39] Duda, R. O., Hart, P. E., & Stork, D. G. (1973). *Pattern classification and scene analysis*, volume 3 : Wiley New York.
- [40] Efron, B. & Hastie, T. (2016). *Computer Age Statistical Inference : Algorithms, Evidence, and Data Science* (1st ed.). : Cambridge University Press.

- [41] El Hajjami, S., Malki, J., Berrada, M., & Fourka, B. (2020). Machine learning for anomaly detection. performance study considering anomaly distribution in an imbalanced dataset. In *2020 5th International Conference on Cloud Computing and Artificial Intelligence : Technologies and Applications (CloudTech)*, (pp. 1–8). IEEE.
- [42] El Hajjami, S., Malki, J., Berrada, M., Harti, M., & Bouju, A. (2021). Machine learning system for fraud detection. a methodological approach for a development platform. In *2021 International Conference on Digital Technologies and Applications (ICDTA)*.
- [43] El Hajjami, S., Malki, J., Bouju, A., & Berrada, M. Machine learning facing behavioral noise problem in an imbalanced data using one side behavioral noise reduction : Application to a fraud detection. *Journal of Computer and Information Engineering*.
- [44] El Hajjami, S., Malki, J., Bouju, A., & Berrada, M. (2020). A machine learning based approach to reduce behavioral noise problem in an imbalanced data : Application to a fraud detection. In *2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, (pp. 11–20). IEEE.
- [45] Estiri, H., Klann, J. G., & Murphy, S. N. (2019). A clustering approach for detecting implausible observation values in electronic health records data. *BMC medical informatics and decision making*, 19(1), 1–16.
- [46] Fan, W., Stolfo, S. J., Zhang, J., & Chan, P. K. (1999). Adacost : misclassification cost-sensitive boosting. In *Icml*, volume 99, (pp. 97–105).
- [47] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). *Learning from imbalanced data sets* : Springer.
- [48] Fernández-Navarro, F., Hervás-Martínez, C., & Gutiérrez, P. A. (2011). A dynamic over-sampling procedure based on sensitivity for multi-class problems. *Pattern Recognition*, 44(8), 1821–1833.
- [49] Freund, Y. & Schapire, R. E. (1996). Schapire R : Experiments with a new boosting algorithm. In *In : Thirteenth International Conference on ML*. Citeseer.
- [50] Friedman, J. H. (2001). Greedy function approximation : a gradient boosting machine. *Annals of statistics*, 1189–1232.
- [51] Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine learning*, 29(2-3), 131–163.
- [52] FTC. Consumer sentinel annual fraud and identity theft reports.
- [53] Fu, K., Cheng, D., Tu, Y., & Zhang, L. (2016). Credit card fraud detection using convolutional neural networks. In *International Conference on Neural Information Processing*, (pp. 483–490). Springer.
- [54] Galar, M., Fernández, A., Barrenechea, E., & Herrera, F. (2013). Eusboost : Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. *Pattern recognition*, 46(12), 3460–3471.

- [55] García, V., Sánchez, J., & Mollineda, R. (2007). An empirical study of the behavior of classifiers on imbalanced and overlapped data sets. (pp. 397–406). Springer Berlin Heidelberg.
- [56] Hart, P. (1968). The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 14(3), 515–516.
- [57] Hartigan, J. A. & Wong, M. A. (1979). A k-means clustering algorithm. *Journal of the Royal Statistical Society : Series C (Applied Statistics)*, 28(1), 100–108.
- [58] Hawkins, D. M. (1980). *Identification of outliers*, volume 11 : Springer.
- [59] Haykin, S. & Network, N. (2004). A comprehensive foundation. *Neural networks*, 2(2004), 41.
- [60] He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). Adasyn : Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, (pp. 1322–1328). IEEE.
- [61] He, H. & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9), 1263–1284.
- [62] Hido, S., Kashima, H., & Takahashi, Y. (2009). Roughly balanced bagging for imbalanced data. *Statistical Analysis and Data Mining : The ASA Data Science Journal*, 2(5-6), 412–426.
- [63] Hu, S., Liang, Y., Ma, L., & He, Y. (2009). Msmote : Improving classification performance when training data is imbalanced. In *2009 second international workshop on computer science and engineering*, volume 2, (pp. 13–17). IEEE.
- [64] Inc, K. (2013). Credit card fraud detection : Anonymized credit card transactions labeled as fraudulent or genuine.
- [65] Jain, A. K. (2010). Data clustering : 50 years beyond k-means. *Pattern recognition letters*, 31(8), 651–666.
- [66] Japkowicz, N. & Stephen, S. (2002). The class imbalance problem : A systematic study. *Intelligent data analysis*, 6(5), 429–449.
- [67] Jeni, L. A., Cohn, J. F., & De La Torre, F. (2013). Facing imbalanced data—recommendations for the use of performance metrics. In *2013 Humaine association conference on affective computing and intelligent interaction*, (pp. 245–251). IEEE.
- [68] Jo, T. & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter*, 6(1), 40–49.
- [69] Joshi, M. V., Kumar, V., & Agarwal, R. C. (2001). Evaluating boosting algorithms to classify rare classes : Comparison and improvements. In *Proceedings 2001 IEEE International Conference on Data Mining*, (pp. 257–264). IEEE.

- [70] Kiran, S., Kumar, N., Guru, J., Katariya, D., Kumar, R., & Sharma, M. (2018). Credit card fraud detection using naïve bayes model based and knn classifier. *International Journal of Advance Research, Ideas and Innovations in Technology*, 4(3).
- [71] Kou, Y., Lu, C.-T., Sirwongwattana, S., & Huang, Y.-P. (2004). Survey of fraud detection techniques. In *IEEE International Conference on Networking, Sensing and Control, 2004*, volume 2, (pp. 749–754). IEEE.
- [72] Koziarski, M., Krawczyk, B., & Woźniak, M. (2019). Radial-based oversampling for noisy imbalanced data classification. *Neurocomputing*, 343, 19–33.
- [73] KPMG. Global banking fraud survey - the multi-faceted threat of fraud : Are banks up to the challenge ?
- [74] Krawczyk, B. (2016). Learning from imbalanced data : open challenges and future directions. *Progress in Artificial Intelligence*, 5(4), 221–232.
- [75] Krawczyk, B., Woźniak, M., & Schaefer, G. (2014). Cost-sensitive decision tree ensembles for effective imbalanced classification. *Applied Soft Computing*, 14, 554–562.
- [76] Krista Tedder, J. B. 2020 identity fraud study : Genesis of the identity fraud crisis.
- [77] Kubat, M. & Matwin, S. (1997). Addressing the curse of imbalanced training sets : one-sided selection. In *Icml*, volume 97, (pp. 179–186). Nashville, USA.
- [78] Lai, Y., Zhang, J., & Liu, Z. (2019). Industrial anomaly detection and attack classification method based on convolutional neural network. *Security and Communication Networks*, 2019.
- [79] Laurikkala, J. (2001). Improving identification of difficult small classes by balancing class distribution. In *Conference on Artificial Intelligence in Medicine in Europe*, (pp. 63–66). Springer.
- [80] Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., & Srivastava, J. (2003). A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the 2003 SIAM international conference on data mining*, (pp. 25–36). SIAM.
- [81] Lepoivre, M. R., Avanzini, C. O., Bignon, G., Legendre, L., & Piwele, A. K. (2016). Credit card fraud detection with unsupervised algorithms. *Journal of Advances in Information Technology*, 7(1).
- [82] Lin, W.-C., Tsai, C.-F., Hu, Y.-H., & Jhang, J.-S. (2017). Clustering-based under-sampling in class-imbalanced data. *Information Sciences*, 409, 17–26.
- [83] Liu, X.-Y., Wu, J., & Zhou, Z.-H. (2008). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2), 539–550.

- [84] López, V., Fernández, A., García, S., Palade, V., & Herrera, F. (2013). An insight into classification with imbalanced data : Empirical results and current trends on using data intrinsic characteristics. *Information sciences*, 250, 113–141.
- [85] Maes, S., Tuyls, K., Vanschoenwinkel, B., & Manderick, B. (2002). Credit card fraud detection using bayesian and neural networks. In *Proceedings of the 1st international naio congress on neuro fuzzy technologies*, (pp. 261–270).
- [86] Menardi, G. & Torelli, N. (2014). Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery*, 28(1), 92–122.
- [87] Mhamane, S. S. & Lobo, L. (2012). Internet banking fraud detection using hmm. In *2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12)*, (pp. 1–4). IEEE.
- [88] Monamo, P., Marivate, V., & Twala, B. (2016). Unsupervised learning for robust bitcoin fraud detection. In *2016 Information Security for South Africa (ISSA)*, (pp. 129–134). IEEE.
- [89] Patidar, R., Sharma, L., et al. (2011). Credit card fraud detection using neural network. *International Journal of Soft Computing and Engineering (IJSCE)*, 1(32-38).
- [90] Phua, C., Lee, V., Smith, K., & Gayler, R. (2010). A comprehensive survey of data mining-based fraud detection research. *arXiv preprint arXiv :1009.6119*.
- [91] Plaibde. Plateforme intégrée big-data pour les données entreprises.
- [92] Prati, R. C., Batista, G. E., & Monard, M. C. (2004). Class imbalances versus class overlapping : an analysis of a learning system behavior. In *Mexican international conference on artificial intelligence*, (pp. 312–321). Springer.
- [93] Priscilla, C. V. & Prabha, D. P. (2019). Credit card fraud detection : A systematic review. In *International Conference on Information, Communication and Computing Technology*, (pp. 290–303). Springer.
- [94] Puh, M. & Brkić, L. (2019). Detecting credit card fraud using selected machine learning algorithms. In *42nd International Convention on Information and Communication Technology, Electronics and Microelectronics*, (pp. 1250–1255). IEEE.
- [95] Quah, J. T. & Sriganesh, M. (2008). Real-time credit card fraud detection using computational intelligence. *Expert systems with applications*, 35(4), 1721–1732.
- [96] Rahman, M. & Davis, D. N. (2013). Cluster based under-sampling for unbalanced cardiovascular data. In *Proceedings of the World Congress on Engineering*, volume 3, (pp. 3–5).
- [97] Rai, A. K. & Dwivedi, R. K. (2020). Fraud detection in credit card data using unsupervised machine learning based scheme. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, (pp. 421–426). IEEE.

- [98] Report (2019). The global payments.
- [99] Roh, Y., Heo, G., & Whang, S. E. (2019). A survey on data collection for machine learning : a big data-ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*.
- [100] Roy, J. & Davenport, M. (2009). Categorization of maritime anomalies for notification and alerting purpose. In *NATO workshop on data fusion and anomaly detection for maritime situational awareness, La Spezia, Italy*, (pp. 15–17).
- [101] Sahin, Y. & Duman, E. (2011). Detecting credit card fraud by ann and logistic regression. In *2011 International Symposium on Innovations in Intelligent Systems and Applications*, (pp. 315–319). IEEE.
- [102] Şahin, Y. G. & Duman, E. (2011). Detecting credit card fraud by decision trees and support vector machines.
- [103] Santos, L. J. S. & Ocampo, S. R. (2018). Bayesian method with clustering algorithm for credit card transaction fraud detection. *Romanian Statistical Review*, (1).
- [104] Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., & Folleco, A. (2014). An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Information Sciences*, 259, 571–595.
- [105] Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., & Napolitano, A. (2009). Rusboost : A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A : Systems and Humans*, 40(1), 185–197.
- [106] Singh, G., Gupta, R., Rastogi, A., Chandel, M. D., & Riyaz, A. (2012). A machine learning approach for detection of fraud based on svm. *International Journal of Scientific Engineering and Technology*, 1(3), 194–198.
- [107] Sun, Y., Kamel, M. S., Wong, A. K., & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12), 3358–3378.
- [108] Sun, Y., Wong, A. K., & Kamel, M. S. (2009). Classification of imbalanced data : A review. *International journal of pattern recognition and artificial intelligence*, 23(04), 687–719.
- [109] Suthaharan, S. (2016). Machine learning models and algorithms for big data classification. *Integr. Ser. Inf. Syst*, 36, 1–12.
- [110] Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics*.
- [111] The Nilson, R. News and statistics for card and mobile payment executives.
- [112] Tomek, I. (1976). An experiment with the edited nearest-neighbor rule.
- [113] Tomek, I. et al. (1976). Two modifications of cnn.

- [114] Vaishali, V. (2014). Fraud detection in credit card by clustering approach. *International Journal of Computer Applications*, 98(3).
- [115] Van Vlasselaer, V., Bravo, C., Caelen, O., Eliassi-Rad, T., Akoglu, L., Snoeck, M., & Baesens, B. (2015). Apaté : A novel approach for automated credit card transaction fraud detection using network-based extensions. *Decision Support Systems*, 75, 38–48.
- [116] Vapnik, V. (2013). *The nature of statistical learning theory* : Springer science & business media.
- [117] Vuttipittayamongkol, P. & Elyan, E. (2020). Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. *Information Sciences*, 509, 47–70.
- [118] Wang, C. & Han, D. (2019). Credit card fraud forecasting model based on clustering analysis and integrated support vector machine. *Cluster Computing*, 22(6), 13861–13866.
- [119] Wang, S. & Yao, X. (2012). Multiclass imbalance problems : Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(4), 1119–1130.
- [120] Wei, Q., Ren, Y., Hou, R., Shi, B., Lo, J. Y., & Carin, L. (2018). Anomaly detection for medical images based on a one-class classification. In *Medical Imaging 2018 : Computer-Aided Diagnosis*, volume 10575, (pp. 105751M). International Society for Optics and Photonics.
- [121] Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, (3), 408–421.
- [122] Xuan, S., Liu, G., Li, Z., Zheng, L., Wang, S., & Jiang, C. (2018). Random forest for credit card fraud detection. In *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, (pp. 1–6). IEEE.
- [123] Zamini, M. & Montazer, G. (2018). Credit card fraud detection using autoencoder based clustering. In *2018 9th International Symposium on Telecommunications (IST)*, (pp. 486–491). IEEE.
- [124] Zaslavsky, V. & Strizhak, A. (2006). Credit card fraud detection using self-organizing maps. *Information and Security*, 18, 48.
- [125] Zhang, C., Bi, J., Xu, S., Ramentol, E., Fan, G., Qiao, B., & Fujita, H. (2019). Multi-imbalance : An open-source software for multi-class imbalance learning. *Knowledge-Based Systems*, 174, 137–143.
- [126] Zheng, A. (2015). Evaluating machine learning models : a beginner’s guide to key concepts and pitfalls.

-
- [127] Zhou, Z.-H. & Liu, X.-Y. (2005). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on knowledge and data engineering*, 18(1), 63–77.
 - [128] Zhu, R., Guo, Y., & Xue, J.-H. (2020). Adjusting the imbalance ratio by the dimensionality of imbalanced data. *Pattern Recognition Letters*.
 - [129] Zhu, T., Lin, Y., & Liu, Y. (2017). Synthetic minority oversampling technique for multiclass imbalance problems. *Pattern Recognition*, 72, 327–340.
 - [130] Zhu, X. & Wu, X. (2004). Class noise vs. attribute noise : A quantitative study. *Artificial intelligence review*, 22(3), 177–210.

Annexe A

Dans le Tableau 1, les différentes versions des composants utilisés sont spécifiées. Certains autres composants de cette distribution qui ne sont pas pertinents pour cette thèse ne sont pas spécifiés ici.

TABLE 1 – Versions utilisées

Package	Version	Description
Scala	2.13.3	Scala (pour Scalable Language) est un langage de programmation multi-paradigme et multi-plateforme très largement utilisé dans le domaine de l'analyse de données et notamment avec le fameux Framework Spark.
Spark	3.0.1	Spark est un cadre de traitement de données rapide dédié au données massives. Il permet d'effectuer un traitement de larges volumes de données de manière distribuée (cluster computing).
Spark.MLlib	3.0	MLlib est la bibliothèque d'apprentissage automatique (ML) de Spark. Son objectif est de rendre l'apprentissage automatique pratique évolutif et facile.
Hadoop	3.3.0	Hadoop est un cadre qui permet le traitement distribué de grands ensembles de données sur des clusters d'ordinateurs à l'aide de modèles de programmation simples.
Yarn	1.22.5	Yarn est gestionnaire de ressource, qui permet de gérer les ressources du système et de planifier les tâches.
JupyterLab	1.1.4	JupyterLab est un environnement de développement interactif basé sur le Web pour les notebooks de Jupyter, le code et les données . JupyterLab est flexible qui prend en charge un large éventail de flux de travail en science des données, en informatique scientifique et en apprentissage automatique. JupyterLab est extensible qui permet d'ajouter de nouveaux composants et s'intégrer aux composants existants.

Annexe B

Une partie du code de cette thèse a été rendue publique dans un référentiel GitHub⁹. Il contient des implémentations qui visent à faciliter le processus de traitement du déséquilibre de classe lors de l'utilisation de Spark ML. Dans la Figure 1, nous pouvons voir un exemple de l'implémentation de l'approche OSBNR pour le prétraitement des données en Scala.

Under-Sampling OSBNR

```
[11]: val s = System.nanoTime
      val newtrainingdata = osbnrClass.Osbnr(trainingData, 2)
      val durationresampling = (System.nanoTime - s) / 1e9d
      println(s"Resampling process takes $durationresampling secs")
      println("New training set statistics: 1 represents Minority class and 0 represents Majority class")
      newtrainingdata.groupBy("Class").count().show()
```

s: Long = 31481665088277861
newtrainingdata: org.apache.spark.sql.DataFrame = [V1: double, V2: double ... 32 more fields]
durationresampling: Double = 27.045339746
Resampling process takes 27.045339746 secs
New training set statistics: 1 represents Minority class and 0 represents Majority class

Class	count
0.0	1944
1.0	393

FIGURE 1 – exemple de l'implémentation de l'approche OSBNR en Scala

9. <https://github.com/salmaelhajjami/>

Annexe C

1 Environnement distribué Spark

1.1 Apache Hadoop

Hadoop¹⁰ -High-availability distributed object-oriented platform- est une plateforme -framework- englobant un certain nombre de services (stockage, analyse et maintenance) permettant de résoudre les problèmes liés au Big Data, open source de la fondation Apache permettant de répondre aux besoins du Big Data à savoir le volume (de l'ordre de plusieurs péta-octets), la variété et la vélocité des données. Il est composé de quatre composants de base :

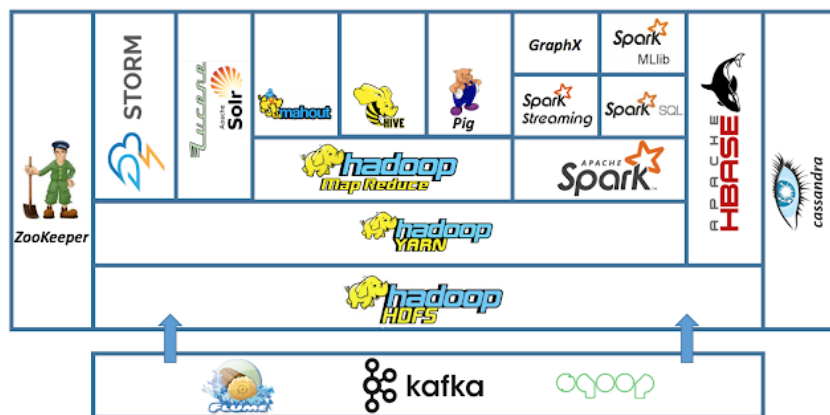


FIGURE 1 – Ecosystème Apache Hadoop.

1. Hadoop Distributed File System (HDFS) : HDFS est un système de fichiers distribué, extensible et portable inspiré par le Google File System (GFS). Il est utilisé principalement pour le stockage persistant de données qui fournit un accès haut-débit aux données de l'application, il a été conçu pour stocker de très gros volumes de données sur un grand nombre de machines équipées de disques durs banalisés. Le HDFS est le composant principal de Hadoop Ecosystème. Il a été conçu pour stocker différents types de grands ensembles de données structurées, non structurées

10. <https://hadoop.apache.org/>

et semi-structurées. Il peut fonctionner dans des machines à faible coût avec une tolérance aux pannes élevée. HDFS définit deux types de nœuds :

- (a) Le nœud principal -NameNode- : Il est caractérisé par :
 - Responsable de la distribution et de la réplication des blocs ;
 - Stocker et gérer les données massive ;
 - Contient la liste des DataNodes pour chaque bloc (dans le cas de l'écriture) ;
 - Contient la liste des blocs pour chaque fichier (dans le cas de lecture).
- (b) Le nœud de données -DataNode- : Il est caractérisé par :
 - Stocker des blocs de données dans le système de fichier local ;
 - Maintenir des métadonnées sur les blocs possédés ;
 - Servir de bloc de données et de métadonnées pour le client HDFS.

Le -NameNode- dans l'architecture Hadoop est un point unique de défaillance. Si ce service est arrêté, il est impossible de pouvoir extraire des blocs d'un fichier donné. Pour résoudre ce problème, un nœud secondaire appelé : Secondary NameNode a été mis en place dans l'architecture Hadoop.

2. Hadoop YARN (Yet Another Resource Negotiator) : Yarn est un framework pour la planification des tâches et la gestion des ressources de calcul en grappes. Il est considéré comme le cerveau de l'écosystème Hadoop. Il rend l'environnement Hadoop mieux adapté aux applications opérationnelles, il effectue toutes les activités de traitement en allouant des ressources et en planifiant des tâches. Hadoop YARN comporte deux composants principaux : Le planificateur -Scheduler- et le gestionnaire d'applications -Applications Manager-. Le planificateur est responsable de l'affectation des ressources aux diverses applications en cours d'exécution ; tandis que, le gestionnaire d'applications est responsable d'accepter les soumissions de travaux et l'exécution d'Application Master.
3. Hadoop Map-Reduce : MapReduce est une structure de programme qui permet d'écrire des applications qui gèrent de grands ensembles de données à l'aide d'algorithmes distribués et parallèles dans l'environnement Hadoop. Un programme MapReduce peut être résumé en deux fonctions Map () et Reduce (). La fonction Map effectue plusieurs actions tels que le filtrage, le regroupement et le tri. Tandis que, La fonction Reduce agrège le résultat produit par la fonction Map. Le résultat généré par la fonction Map est une paire de valeurs de clé (K, V) qui sert d'entrée à la fonction de réduction.
4. Hadoop Common : Hadoop Common contient toutes les bibliothèques et les outils nécessaires pour que d'autres logiciels puissent se servir d'Hadoop. Plus concrètement, il permet de couvrir le stockage et la répartition des données, les traitements distribués, l'entrepôt de données, le workflow, la programmation, sans oublier la coordination de l'ensemble des composants.

1.2 Apache Spark

Apache Spark¹¹ est un framework d'analyse de données en temps réel dans un système distribué, et il est récemment devenu plus populaire que Hadoop dans la technologie Big Data. Il exécute des calculs en mémoire pour augmenter la vitesse de traitement des données sur Map-Reduce. Il est plus rapide que Hadoop MapReduce pour le traitement de données à grande échelle en exploitant des calculs en mémoire et d'autres optimisations. Spark présente plusieurs avantages par rapport aux autres technologies Big Data notamment MapReduce du Hadoop et Storm. D'abord, Spark propose un framework complet et unifié pour répondre aux besoins de traitements Big Data ainsi que par type de traitement (batch, micro batch, ou streaming). Ensuite, Spark permet à des applications sur Hadoop d'être exécutées jusqu'à 100 fois plus rapide en mémoire et 10 fois plus rapide sur disque. Il nous permet d'écrire rapidement des applications en Java, Scala, R ou Python, il inclut également un jeu de plus de 80 opérateurs haut-niveau. Il est composé de cinq composants de base :

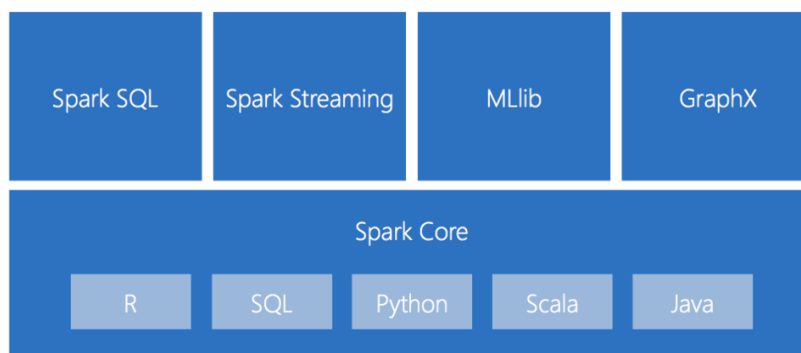


FIGURE 2 – Ecosystème Apache Spark.

1. Spark SQL : Spark SQL est un module Spark pour le traitement de données structurées. Il fournit une abstraction de programmation appelée -DataFrames-, il peut également agir en tant que moteur de requête SQL distribué. Il permet aux requêtes Hadoop Hive non modifiées de s'exécuter jusqu'à 100 fois plus rapidement sur les déploiements et les données existants. Il fournit également une intégration puissante avec le reste de l'écosystème Spark (par exemple, en intégrant le traitement de requête SQL avec l'apprentissage automatique).
2. Spark Stearming : Spark Streaming permet de puissantes applications interactives et analytiques pour Big Data en héritant des caractéristiques de facilité d'utilisation et de tolérance aux pannes de Spark. Il s'intègre facilement à une grande variété de sources de données populaires, notamment HDFS, Flume et Kafka. Spark Streaming peut être utilisé pour les traitements des données en temps-réel, il utilise le module DStream, c-à-d une série de RDD -Resilient Distributed Dataset-.

11. <https://spark.apache.org/>

3. Spark MLlib : MLlib est une bibliothèque d'apprentissage automatique -Machine Learning- permettant d'analyser le Big Data, il fournit plus de 55 algorithmes d'apprentissage automatique évolutifs qui bénéficient largement de la parallélisation des données. Il fournit également à la fois des algorithmes de haute qualité (par exemple, plusieurs itérations pour augmenter la précision) et une vitesse fulgurante (jusqu'à 100 fois plus rapide que MapReduce). MLlib du Spark est utilisé avec des applications écrites en Java, Scala, R et Python afin que nous puissions l'inclure dans le traitement du Big Data.
4. Spark GraphX : GraphX est un moteur de calcul graphique qui permet la composition de graphes à partir des données non structurées et des tableaux, il permet également d'afficher les mêmes données physiques sous forme de graphe, sans déplacement ni duplication de données.
5. Spark Core : Spark Core est le moteur d'exécution général sous-jacent de la plateforme Spark sur lequel toutes les autres fonctionnalités sont créées. Spark Core utilise une structure de données fondamentale spécialisée : RDD, RDD est un ensemble logique de données partitionnées sur plusieurs ordinateurs.
Les RDD peuvent être créés de deux manières : la première consiste à référencer des ensembles de données dans des systèmes de stockage externes, et la seconde consiste à appliquer des transformations (par exemple, carte, filtre, réducteur, jointure) aux RDD existants.

Différentes options de gestion de cluster sont disponibles pour exécuter Spark : elles vont du simple planificateur autonome intégré de Spark à d'autres gestionnaires de cluster répandus tels que Apache Mesos et Hadoop YARN.



Centre d'Etudes Doctorales : Sciences et Techniques de l'Ingénieur

Titre de la thèse : Élaboration d'une technique d'apprentissage automatique pour la détection de fraude par carte de crédit : Application sur les données massives déséquilibrées.

Nom et prénom du candidat : EL HAJJAMI Salma

Spécialité : Informatique

Résumé de la thèse

A l'ère numérique d'aujourd'hui, les cartes de crédit sont utilisées dans pratiquement toutes les transactions. La facilité de paiement en ligne au cours de la dernière décennie, a ouvert une nouvelle passerelle pour les opportunités, les entreprises, l'abaissement des frontières géographiques, mais aussi l'augmentation des transactions frauduleuses. Alors que le pic du graphique du commerce électronique ne cesse d'atteindre de nouveaux sommets chaque année, il est également devenu la toile de fond des fraudeurs qui abusent fortement de la transparence des méthodes de paiement en ligne et des transactions par carte de crédit. Par conséquent, la détection de fraudes est un aspect essentiel de l'identification des transactions frauduleuses qui ont été émises illégalement au nom du propriétaire légitime de la carte et de la correction des pertes.

Les progrès récents des approches d'intelligence artificielle, comme l'apprentissage automatique ou Machine Learning (ML), font une percée importante dans ce domaine grâce à leur travail en temps réel, leur vitesse et leur efficacité. La détection de fraudes qui était autrefois un domaine qui nécessitait beaucoup de travail manuel, est maintenant l'un des nombreux à avoir connu les progrès grâce à la capacité des algorithmes de ML à apprendre des modèles de fraude historiques et à les reconnaître dans les transactions futures. Néanmoins, la conception d'algorithmes de détection de fraude est sujette à divers problèmes, qui ont un impact considérable sur la précision du système.

Le travail de recherche présenté dans cette thèse concerne l'élaboration d'une technique à base d'apprentissage automatique pour la détection de fraudes. Nous traitons un problème majeur rencontré par les méthodes d'apprentissage dans la détection de fraudes qui concerne la distribution asymétrique des classes, également connue sous le nom de déséquilibre de classes, qui se produit lorsqu'une des classes a plus d'instances qu'une autre classe. C'est particulièrement le cas du domaine de détection de fraudes, en effet nous rencontrons souvent des transactions frauduleuses minoritaires qui sont mal représentées lors de la phase d'apprentissage, ce qui perturbe les algorithmes ML qui présentent une grande précision sur les instances normales mais une faible précision sur les instances frauduleuses.

Afin d'apporter des éléments de réponse pour cette problématique, nous proposons différentes contributions. Nous avons fourni comme première contribution, une étude expérimentale de l'influence du déséquilibre de classes sur les performances des algorithmes d'apprentissage automatique pour comprendre les véritables faiblesses. Dans une seconde contribution, nous avons proposé une méthode qui agit au niveau des données d'apprentissage appelée : One Side Behavioral Noise Reduction (OSBNR). C'est une technique d'échantillonnage qui permet de rééquilibrer les données en sous-échantillonnant la classe majoritaire. Enfin, nous avons proposé une méthodologie générale basée sur la formalisation, la mise en œuvre et la conception technique d'une plateforme de détection de fraudes en mesure de satisfaire les conditions de travail dans le monde réel.

Mots clés : Détection de fraude, Détection d'anomalie, Apprentissage automatique, Déséquilibre de classe, Big Data, Méthodes d'échantillonnage, Plates-formes Spark et Hadoop.