République du Bénin

****

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique (MESRS)

****

Université d'Abomey-Calavi (UAC)

****

Ecole Polytechnique d'Abomey-Calavi (EPAC)

****

Département de Génie Informatique et Télécommunications (GIT)

****

Option : Réseaux Informatiques et Internet (RII)

Mémoire de fin de formation pour l'obtention du diplôme d'ingénieur de conception grade master

Thème :

Outil automatisé de pré-traitement éthique et transparent des données.

Réalisé et soutenu par :

BONOU Mael Habib Sedjro

Maître de mémoire :

Dr SOGBOHOSSOU Medesu

Enseignant - Chercheur à l'Ecole Polytechnique d'Abomey-Calavi.

Encadreur :

Dr (MA) HOUNDJI Ratheil

Enseignant - Chercheur à l'IFRI

Année académique : 2023 - 2024

République du Bénin

****

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique (MESRS)

****

Université d'Abomey-Calavi (UAC)

****

Ecole Polytechnique d'Abomey-Calavi (EPAC)

****

Département de Génie Informatique et Télécommunications (GIT)

****

Directeur

Docteur ALITONOU Guy Alain
(Professeur Titulaire des Universités du CAMES)

Directeur Adjoint

(Chargé des affaires académiques)

PRODJINONTO Vincent
(Professeur Titulaire des Universités du CAMES)

Chef de département

Docteur DJARA Tahirou
(Maître de Conférences des Universités du CAMES)

Année académique : 2023 - 2024

17ème Promotion

LISTE DES ENSEIGNANTS AYANT INTERVENU DANS NOTRE FORMATION DE 2019 A 2024

NOM	**PRÉNOMS**	**MATIÈRES ENSEIGNÉES**
ABALLO	Théophile	Réseaux Locaux, Conception des Systèmes d'Information (UML), Protocoles : Couches de Connexion de Réseaux et de Communication
ADANHOUNME	Villévo	Analyse et calcul intégral, Analyse vectorielle et Équation différentielle
AGBAHUNGBA	Georges	Méthodologie de Recherche Documentaire
AGBOMAHENAN	Macaire	Électromagnétisme, Technologie des Mesures Électriques
AKOWANOU	Onésime	Électrochimie
ALLOGNON	Elisabeth	Cinématique et Dynamique
ANJORIN	Malahimi	Thermodynamique, Transferts Thermiques
ASSOGBA	Emery	Architecture des ordinateurs, Microprocesseurs et microcontrôleurs
ASSOGBA	Marc	Probabilité et processus aléatoires, Programmation JAVA, Traitement Numérique de la Parole
BOURAIMA	Marcos	Recherche Opérationnelle
BOCCO	Elvarez	Algorithme numérique
CHETANGNY	Patrice	Compatibilité Électromagnétique
CHITOU	Naimoulai	Dessin Technique

NOM	PRÉNOMS	MATIÈRES ENSEIGNÉES
CODJIA	Wilfried	Gestion des Projets Informatiques
CODO	Paul	Statique Graphique et Analytique
COMLAN	Maurice	Programmation Langage C++
DEGAN	Gérard	Analyse Numérique
DEGBO	Basile	Transmission analogique, Rayonnement et antenne, Systèmes et Technologies Radio mobile et sans fil
DEGUENON	Judicaël	Probabilité et Statistique, Équation aux dérivées partielles
DJARA	Tahirou	Systèmes de Gestion de Base de Données, Environnement et développement logiciel, Traitement Numérique de l'Image
DJOGBE	Léopold	Fonctions Électroniques, Transmission Numérique du signal, Circuits radio fréquences et Dispositifs hyperfréquences
DOGUE	Karel	Droit Industriel
DOSSOU	Michel	Introduction Générale aux Télécommunications, Architecture des Réseaux, Dispositifs et transmission optique, Télévision Numérique Terrestre
EGOUNLETY	Richard	Physique des semi-conducteurs et applications
GBAGUIDI	Gérard	Résistance des Matériaux
GUEDJE	François	Initiation à l'algorithmique, Langage et Programmation Informatique
HOUANOU	Agapi	Analyse tensorielle
HOUDEDAKO	Vincent	Circuits Électriques

NOM	PRÉNOMS	MATIÈRES ENSEIGNÉES
HOUNDJI	Ratheil	Intelligence Artificielle et Applications
HOUNGAN	Théophile	Signaux et Systèmes
JOSSOU	Thierry	Électricité générale
MONTEIRO	Léonard	Convertisseurs numériques et circuits logiques programmables
LALEYE	Claude	Base et Contrôle de Qualité
NOUNAGNON	Charles	Métrologie et analyse des données
N'TCHA	Sabi	Outils Statistiques en Milieu Industriel
PRODJINONTO	Vincent	Mécanique des fluides
SANYA	Max Fréjus	Circuits Logiques Combinatoires et Séquentiels, Théorie des Graphes et Réseaux, Réseaux de Télécommunications, Micro ondes et Radar
SEMASSOU	Clarence	Optique Géométrique
SEGUEDEME	Alexis	Anglais Technique V et VI
SEWANOUDE	Damien	Comptabilité Générale, Création et Gestion des Entreprises, Marketing
SOEDE	Casimir	Anglais Technique I, II, III et IV
SOGBOHOSSOU	Mèdésu	Système d'Exploitation, Langage C, Théorie des Langages et Interpréteurs, Systèmes Temps Réel et Systèmes Embarqués
SOHOUNHLOUE	Dominique	Chimie pour l'ingénieur
SOTINDJO	Patrick	Traitement numérique du signal, Gestion du spectre de fréquences
TOGNIVO	Marc	Éducation Physique et Sportive
ZOUTANGNI	Laurent	Techniques d'Expression et Méthodes de Communications I et II

DÉDICACE

REMERCIEMENTS

Mes sincères remerciements :

LISTE DES SIGLES ET ABRÉVIATIONS

A

B

C

D

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SYMBOLES ET UNITÉS DE MESURES

RÉSUMÉ

ABSTRACT

SOMMAIRE

INTRODUCTION GÉNÉRALE

Dans un monde de plus en plus axé sur les données, la capacité à traiter et à analyser de grandes quantités d'informations est devenue un atout stratégique majeur pour les entreprises, les gouvernements et les chercheurs. Les données sont aujourd'hui au cœur des décisions stratégiques, des innovations technologiques et des avancées scientifiques. Cependant, l'intégrité et la qualité des données sont souvent compromises par des erreurs, des biais, et des informations incomplètes. Le prétraitement des données, qui englobe des étapes telles que le nettoyage, la normalisation et la détection des biais, est essentiel pour garantir des analyses fiables et des décisions éclairées.

Historiquement, le prétraitement des données a été effectué manuellement, ce qui présente des risques d'erreurs humaines et nécessite un temps considérable. Avec l'essor du Big Data et de l'intelligence artificielle (IA), plusieurs études ont tenté d'automatiser ces étapes. Par exemple, l'introduction des bibliothèques telles que Pandas et Scikit-learn a permis de faciliter le nettoyage et la préparation des données. Cependant, ces solutions, bien que fonctionnelles, manquent souvent de mécanismes intégrés pour détecter et mitiger les biais présents dans les ensembles de données, posant ainsi des défis éthiques majeurs [1].

Aujourd'hui, l'éthique dans le traitement des données est devenue une préoccupation centrale. Des recherches récentes, telles que celles sur les biais algorithmiques, ont mis en évidence le danger que représentent les biais inconscients dans les systèmes automatisés. Des solutions comme AIF360 d'IBM, les Indi-

cateurs d'équité de Google et Fairlearn ont été développées pour détecter ces biais, mais elles restent limitées en termes de couverture et de transparence. Peu d'outils automatisés intègrent à la fois le prétraitement des données et une analyse exhaustive des biais, tout en générant des rapports détaillés et explicatifs pour assurer la transparence du processus.

C'est dans ce contexte que se pose la problématique de cette étude : **comment développer un outil automatisé de prétraitement des données qui soit à la fois éthique et transparent, tout en répondant aux besoins actuels des acteurs de la Data Science ?** Si des progrès notables ont été réalisés dans l'automatisation du prétraitement, il reste des lacunes importantes dans l'intégration de principes éthiques et de mécanismes de transparence, essentiels pour garantir des analyses justes et impartiales.

Le but de cette étude est donc de développer un outil automatisé qui intègre non seulement les étapes classiques de prétraitement des données (nettoyage, réduction de dimensionnalité), mais aussi des fonctionnalités permettant de détecter et mitiger les biais potentiels, tout en générant des rapports explicatifs détaillés des actions entreprises. Cet outil vise à répondre aux besoins croissants de transparence et d'éthique dans le traitement des données, en fournissant aux utilisateurs des recommandations claires et justifiées, contribuant ainsi à une meilleure prise de décision basée sur des données fiables et impartiales.

CONTEXTE, JUSTIFICATION ET PROBLÉMATIQUE

Le prétraitement des données inclut des étapes comme le nettoyage, la réduction de dimensionnalité et la détection des biais, souvent effectuées manuellement, ce qui les rend fastidieuses et sujettes aux erreurs. Le manque de transparence pose également des préoccupations éthiques liées à la discrimination et aux biais implicites. L'utilisation d'algorithmes avancés, l'apprentissage automatique ainsi que l'intelligence artificielle générative, peut permettre d'automatiser ces tâches tout en garantissant transparence et éthique.

Le développement d'un outil automatisé de prétraitement des données, comme proposé dans ce mémoire, répond à un besoin croissant dans divers domaines, tels que la recherche scientifique, les entreprises et les institutions gouvernementales [2]. En intégrant des librairies de données éthiques et des analyses de corrélation, cet outil vise à améliorer la qualité des données et à garantir des analyses plus fiables et impartiales. De plus, il permettra un gain de temps considérable dans le processus de développement de modèle d'intelligence artificielle, en raison de sa grande simplicité d'utilisation.

La justification de ce projet repose sur l'importance de disposer de données de haute qualité pour des analyses précises et des décisions éclairées, et ce, avec un effort et un temps minimum. En automatisant le prétraitement des données et en assurant une transparence totale, cet outil contribuera à renforcer la confiance dans les analyses de données, à promouvoir des pratiques éthiques dans la gestion des données et à réduire le temps de développement.

HYPOTHÈSES DE RECHERCHE

— L'intégration d'un outil automatisé de prétraitement des données, doté de mécanismes de détection et de mitigation des biais, améliore l'éthique dans le traitement des données.

— La génération de rapports détaillés et auditables sur chaque étape du prétraitement renforce la traçabilité des actions entreprises et réduit les erreurs humaines.

— L'automatisation du prétraitement des données optimise le processus de préparation des données pour des analyses plus rapides, plus fiables et impartiales.

OBJECTIFS DE L'ÉTUDE

L'objectif général de cette étude est de développer un outil automatisé de prétraitement des données qui garantit une approche éthique et transparente en inté-

grant des mécanismes de détection des biais, de génération de rapports détaillés, et d'audit complet des étapes de prétraitement. Il s'agit spécifiquement de :

1. Identifier et classifier automatiquement les types de données (quantitatives, qualitatives, et démographiques) afin de faciliter le prétraitement initial et la détection des biais potentiels.
2. Automatiser le nettoyage des données et la réduction de dimensionnalité, tout en intégrant des algorithmes de détection et de mitigation des biais, pour assurer la qualité des données.
3. Générer des rapports détaillés et auditables expliquant chaque étape du prétraitement, les techniques utilisées, et les actions de mitigation des biais entreprises, tout en offrant un audit du code pour vérifier sa conformité éthique.

Les résultats attendus sont :

1. Résultat attendu : Une classification précise des données pour chaque ensemble de données analysé, permettant d'orienter le processus de nettoyage et d'analyse des biais.
2. Résultat attendu : Des ensembles de données propres, normalisés, et débarrassés de biais explicites, prêts pour l'analyse.
3. Résultat attendu : Des rapports compréhensibles et exploitables, documentant les actions entreprises pour garantir la transparence et la traçabilité de toutes les opérations de prétraitement.

Première partie

SYNTHÈSE BIBLIOGRAPHIQUE

Chapitre 1

AUTOMATISATION DU PRÉTRAITEMENT DES DONNÉES

Le prétraitement des données, crucial pour l'analyse, est traditionnellement manuel et sujet à des erreurs. Avec l'essor du Big Data, l'automatisation a amélioré l'efficacité et la précision de ces tâches comme le nettoyage et la réduction de dimensionnalité.

Dans ce premier chapitre, nous allons explorer dans un premier temps le concept d'automatisation du pretraitement des données. Ensuite nous aborderons le sujet des considérations éthiques dans le traitement automatisé des données ainsi que de la transparence dans les outils automatisés.

Enfin, nous ferons un état de l'art des realisations qui ont déjà été faites dans le cadre de notre études.

L'automatisation du prétraitement des données est une étape critique dans les projets d'analyse de données, notamment dans les contextes de Big Data. Le prétraitement est nécessaire pour garantir la qualité et l'exploitabilité des données, mais il est également connu pour être une phase chronophage, représentant entre 50 et 80% du temps consacré à un projet d'analyse [3]. Sans une préparation adéquate des données, même les meilleurs modèles d'apprentissage automatique risquent de produire des résultats imprécis [1]. De plus, les données collectées à partir de n'importe quelle source restent incomplètes, bruyantes et incohérentes, ce qui entraîne des problèmes dans l'analyse des données. Il est donc nécessaire de rectifier les problèmes à l'avance. Ces problèmes peuvent

être classés en trois groupes : trop de données, trop peu de données et des données fragmentées. La figure suivante explique les problèmes posés par les données sous forme de tableaux [3].

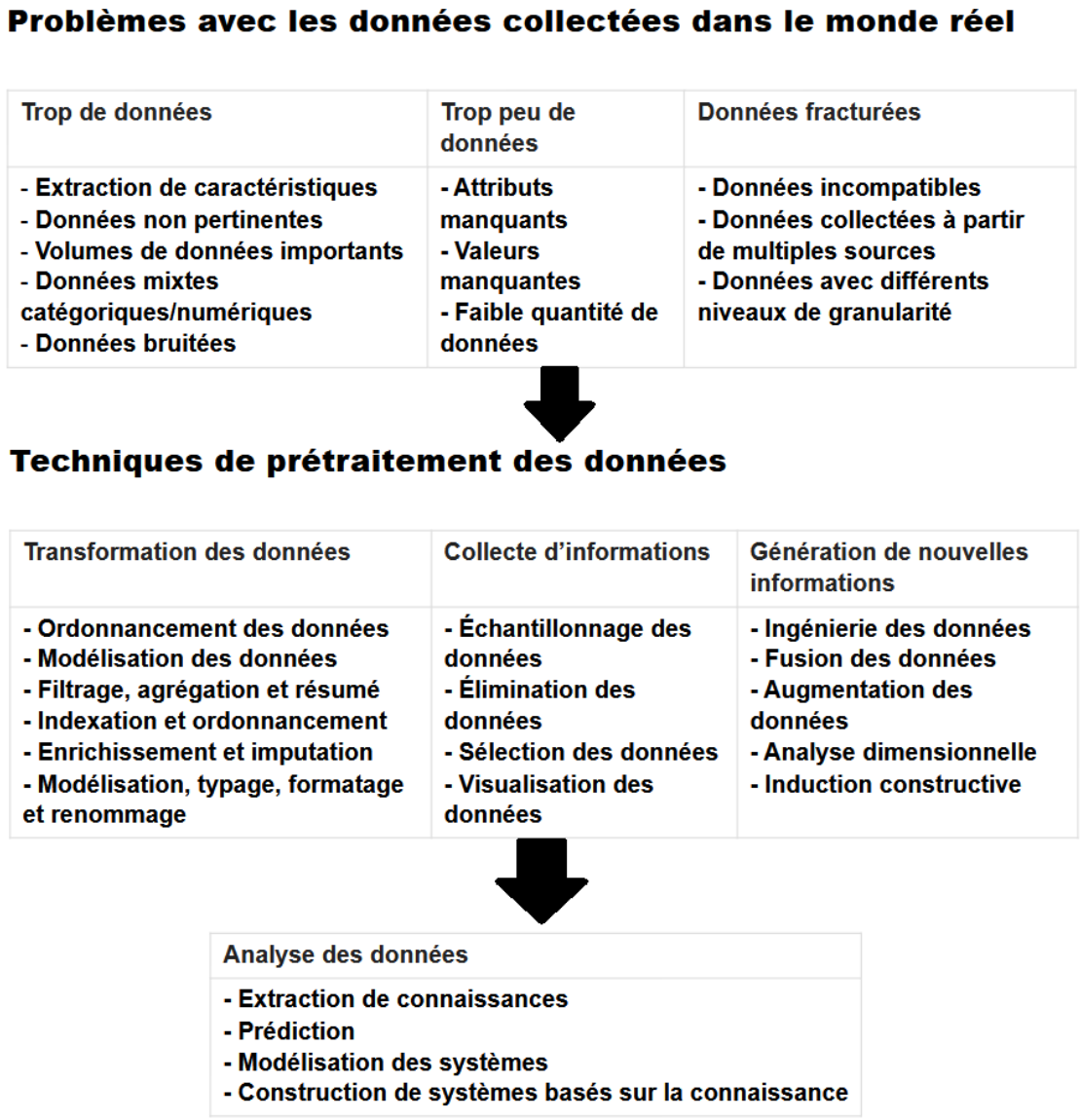

Figure 1.1 – Problèmes liés aux données collectées dans le monde réel. [3]

1.1 Clarification conceptuelle

Le prétraitement des données est une étape cruciale dans les pipelines d'apprentissage automatique et d'Intelligence Artificielle (AI). Il englobe des opérations telles que le nettoyage des données, la gestion des valeurs manquantes, la détection et la correction des anomalies, ainsi que la normalisation et la réduction de dimensionnalité. Ces processus sont essentiels pour préparer les données à une analyse efficace et pour garantir la qualité des résultats obtenus. Historiquement, ces tâches étaient réalisées manuellement, ce qui était à la fois chronophage et sujet à des erreurs humaines. Cependant, avec l'avènement des outils modernes comme Pandas et Scikit-learn, une grande partie de ces étapes a été automatisée, permettant ainsi de gagner en efficacité et en précision.

L'approche moderne de l'AI vise à concevoir des algorithmes capables d'apprendre directement à partir des données. Cette approche a permis des avancées significatives, notamment dans le domaine de l'apprentissage supervisé et de l'apprentissage profond (deep learning). Cependant, malgré ces progrès, toutes les étapes de traitement des données dans les pipelines d'apprentissage automatique n'ont pas été entièrement automatisées. Traditionnellement, les données doivent être collectées manuellement, prétraitées et enrichies par des techniques d'augmentation de données avant de pouvoir être utilisées pour l'entraînement des modèles. Récemment, des techniques spécialisées pour automatiser ces tâches ont émergé, répondant au besoin croissant de traiter de grands volumes de données complexes et hétérogènes pour les applications de big data et d'apprentissage automatique.

Aujourd'hui, des systèmes de traitement de données end-to-end basés sur des techniques d'apprentissage automatique automatisé (Automated Machine Learning (AutoML)) sont capables de transformer des données brutes en caractéristiques utiles pour des tâches de big data, en automatisant toutes les étapes intermédiaires. Cela inclut le prétraitement automatisé (nettoyage des données, imputation des valeurs manquantes, encodage des données catégorielles), l'augmentation des données (y compris la génération de données synthétiques via des méthodes d'IA générative) et l'ingénierie des caractéristiques (extraction,

construction et sélection automatisées des caractéristiques). Ces systèmes permettent non seulement d'automatiser des tâches spécifiques, mais aussi d'optimiser simultanément toutes les étapes du pipeline d'apprentissage automatique.

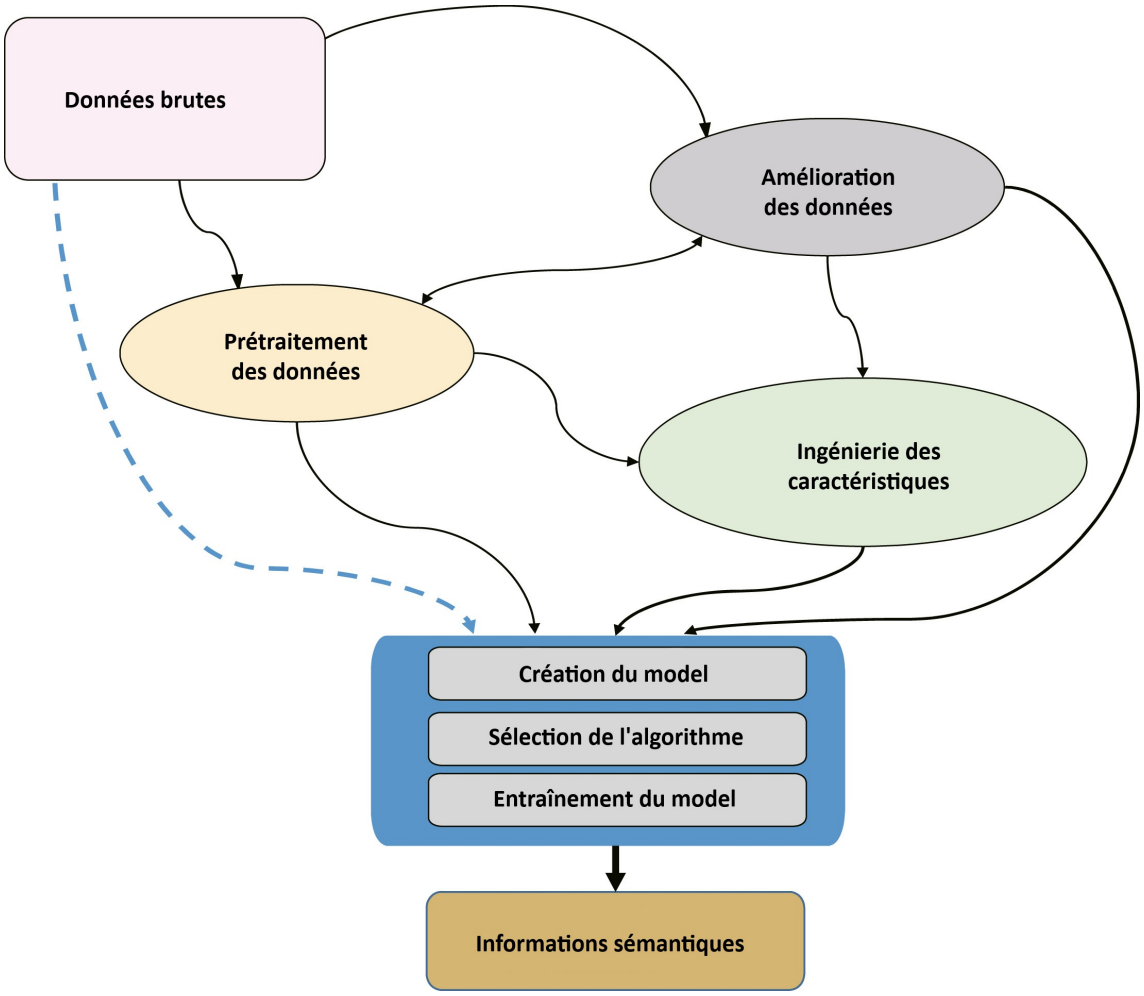

Figure 1.2 – Flux de travail de base de la préparation des données et de l'ingénierie des caractéristiques dans l'apprentissage automatique et le développement d'applications big data. [4]

En résumé, l'automatisation du prétraitement des données est motivée par la nécessité de gérer des volumes croissants de données complexes tout en garantissant leur qualité et leur utilité pour les modèles d'AI. Les outils modernes et les techniques d'AutoML jouent un rôle clé dans cette évolution, rendant les pipelines de traitement des données plus efficaces, transparents et adaptés aux défis actuels du big data et de l'apprentissage profond [1].

1.2 Solutions existantes

Aujourd'hui, plusieurs solutions logicielles, comme H2O-AutoML, AutoSklearn et DataRobot, intègrent des fonctions de prétraitement automatisé. Ces outils sont capables de détecter les types de données (numériques, catégorielles, etc.) et de réaliser certaines transformations de base. Cependant, la majorité de ces outils nécessitent une intervention humaine pour les configurations plus avancées, telles que la sélection des méthodes de transformation adaptées aux besoins spécifiques des utilisateurs. L'une des principales lacunes de ces solutions réside dans leur incapacité à proposer des recommandations pertinentes pour choisir les meilleurs paramètres de transformation, ce qui peut affecter la qualité des résultats obtenus[4].

1.2.1 Description et comparaison de quelques outils

1.2.2 Google AutoML

— **Description** : Google AutoML est une plateforme cloud qui automatise la création de modèles de machine learning, y compris le prétraitement des données. Il est conçu pour les utilisateurs ayant peu d'expérience en machine learning.

— **Points forts** :

  — Automatisation complète du prétraitement (nettoyage, encodage, normalisation).

  — Intégration avec les services Google Cloud.

  — Support pour les données structurées et non structurées (texte, images).

Figure 1.3 – Google AutoML [1]

1.2.3 H2O.ai

Description

H2O.ai est une plateforme open-source et commerciale qui automatise l'ensemble du pipeline de machine learning, y compris le prétraitement des données. Elle est connue pour son framework H2O et son produit Driverless AI.

Points forts

— Automatisation avancée du prétraitement (gestion des valeurs manquantes, sélection de caractéristiques).

— Ingénierie des caractéristiques automatisée.

— Détection des biais et explication des modèles.

Figure 1.4 – H2O.ai [2]

1.2.4 DataRobot

Description

DataRobot est une plateforme d'apprentissage automatique automatisé qui permet de créer, déployer et gérer des modèles de machine learning. Elle automatise également le prétraitement des données.

Points forts

— Nettoyage et transformation automatiques des données.

— Sélection et ingénierie des caractéristiques.

— Détection des biais et explication des modèles.

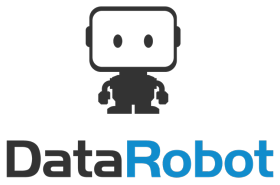

Figure 1.5 – DataRobot [3]

1.2.5 Trifacta Wrangler (DataWrangler)

Description

Trifacta Wrangler est un outil de préparation de données qui utilise l'IA pour automatiser le nettoyage et la transformation des données. Il est conçu pour les utilisateurs techniques et non techniques.

Points forts

— Interface visuelle intuitive pour le prétraitement des données.

— Détection automatique des schémas de données.

— Transformation automatisée des données.

Figure 1.6 – Trifacta Wrangler (DataWrangler) [4]

1.2.6 IBM Watson Studio (AutoAI)

Description

International Business Machines (IBM) Watson Studio est une plateforme cloud qui inclut une fonctionnalité appelée AutoAI pour automatiser le prétraitement des données et la construction de modèles.

Points forts

— Nettoyage et transformation automatiques des données.

— Sélection et ingénierie des caractéristiques.

— Détection des biais et explication des modèles.

Figure 1.7 – IBM Watson Studio (AutoAI) [5]

1.2.7 Tableau Comparatif

Tableau I – Comparaison des outils de prétraitement automatisé

Outils	**Google AutoML**	**H2O.ai**	**DataRobot**	**Trifacta Wrangler**	**IBM Watso Studio (Auto**
Niveau d'intervention	Très faible	Très faible	Très faible	Faible à modéré	Très faible
Gestion éthique mais limitée correction transparence gestion manuelle transparence	Détection des biais, Détection et Détection + Qualité des données, Détection +				
Automatisation	Complète	Complète	Complète	Partielle	Complète
Support données non structurées non structurées	Structurées et Structurées	Structurées	Structurées	Structurées et	
Cloud	Google	Multi	Multi	Multi	IBM
Coût + payant	Payant Payant	Open-source Payant	Payant		
Facilité	Très facile	Modérée	Facile	Très facile	Facile

1.3 Analyse des Axes de Comparaison

Niveau d'intervention humaine nécessaire

— Google AutoML, H2O.ai, DataRobot, et IBM Watson Studio nécessitent un **niveau d'intervention humaine très faible**. Ces outils automatisent presque entièrement le prétraitement des données.

— Trifacta Wrangler nécessite un niveau d'intervention faible à modéré en raison de son interface visuelle, qui permet à l'utilisateur de valider les transformations.

Gestion de l'aspect éthique

— H2O.ai, DataRobot, et IBM Watson Studio offrent des fonctionnalités avancées pour la **détection des biais** et la **transparence des modèles**.

— Google AutoML propose des outils pour détecter les biais, mais la gestion éthique dépend davantage de l'utilisateur.

— Trifacta Wrangler se concentre sur la qualité des données, mais la gestion des biais et des aspects éthiques est moins automatisée.

Chapitre 2

CONSIDERATIONS ETHIQUES DANS LE TRAITEMENT AUTOMATISE DES DONNEES

2.1 Clarification conceptuelle

L'essor de l'intelligence artificielle (IA) et des technologies de traitement automatisé des données a transformé la manière dont les organisations prennent des décisions, interagissent avec leurs clients et gèrent leurs opérations. Cependant, cette révolution technologique s'accompagne de défis éthiques majeurs, notamment en ce qui concerne la transparence, la responsabilité, et l'équité des systèmes automatisés. Alors que les entreprises adoptent de plus en plus ces technologies pour améliorer leur efficacité et leur compétitivité, les questions éthiques liées à l'utilisation des données et à l'automatisation des processus deviennent centrales.

Les systèmes de traitement automatisé des données, bien que puissants, peuvent introduire des biais, compromettre la vie privée des individus, et générer des décisions injustes ou discriminatoires si elles ne sont pas correctement conçues et surveillées. Par exemple, des algorithmes d'IA utilisés pour le recrutement ont montré des préjugés sexistes en favorisant les candidats masculins, reflétant les biais présents dans les données historiques utilisées pour leur entraînement[5]. Ces exemples soulignent la nécessité de mettre en place des cadres éthiques robustes pour guider le développement et l'utilisation des technologies de trai-

tement automatisé des données.

Dans ce contexte, il est essentiel de clarifier les concepts clés liés à l'éthique dans le traitement automatisé des données et d'explorer les principes qui peuvent guider une utilisation responsable de ces technologies. Cette section vise à fournir une compréhension approfondie des enjeux éthiques associés à l'automatisation du traitement des données, en mettant l'accent sur la responsabilité des données, l'alignement des valeurs et la redevabilité algorithmique.

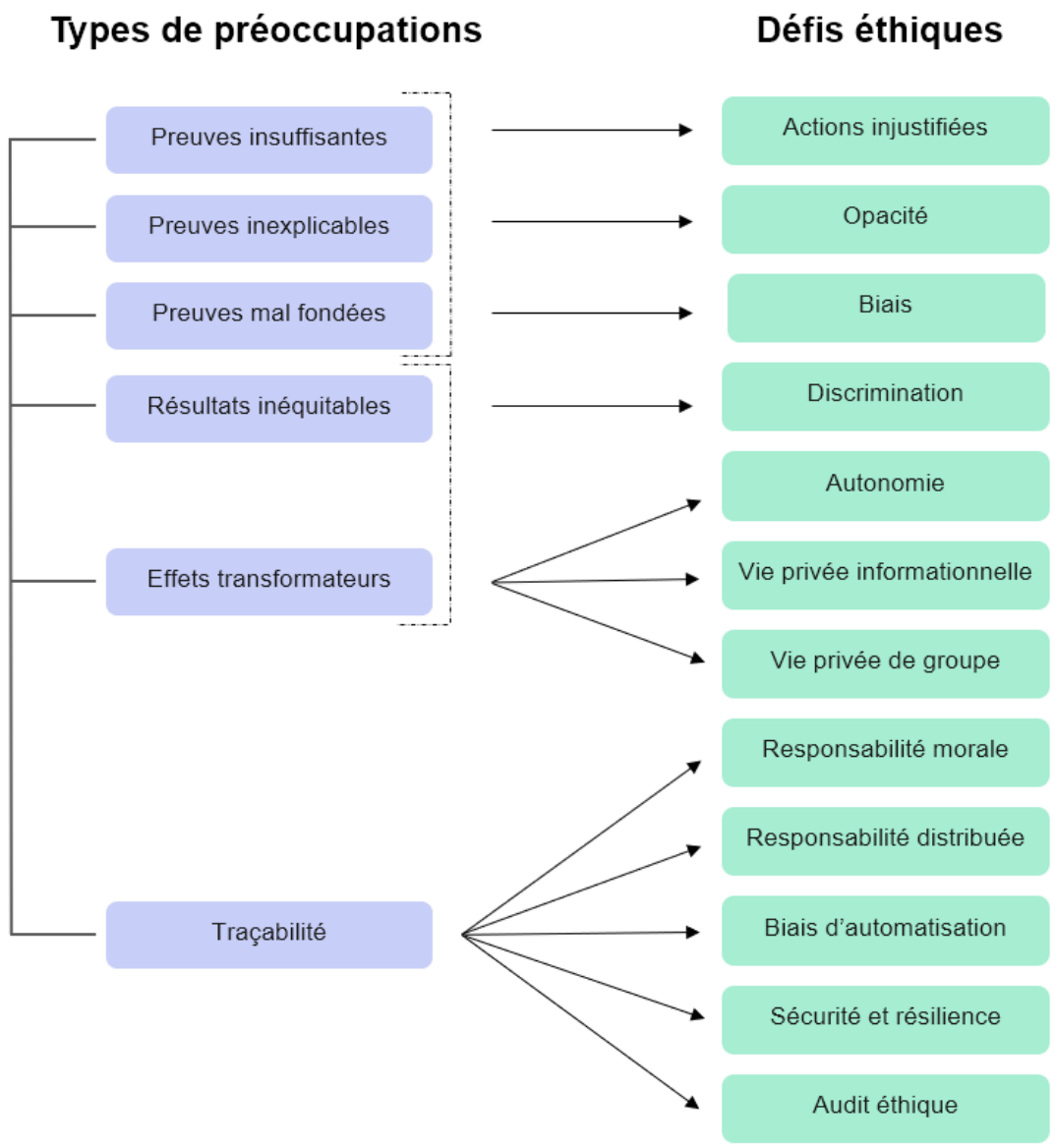

Figure 2.1 – Types de préoccupations et défis liées à l'éthique de l'IA [**?**]

2.1.1 Responsabilité des données

La responsabilité des données est un pilier central de l'éthique dans le traitement automatisé des données. Elle englobe la gestion éthique des données tout au long de leur cycle de vie, depuis leur collecte jusqu'à leur stockage, leur utilisation et leur partage. Les organisations doivent garantir que les données sont utilisées de manière transparente, sécurisée et respectueuse de la vie privée des individus.

Cela inclut la mise en place de mesures pour prévenir les violations de données, assurer la confidentialité, et obtenir le consentement éclairé des utilisateurs.

Selon une étude de l'IBM Institute for Business Value, 81 pour cent des consommateurs ont exprimé une préoccupation accrue quant à l'utilisation de leurs données par les entreprises, et 75% ont déclaré être moins enclins à faire confiance aux organisations pour la gestion de leurs informations personnelles. Ces chiffres soulignent l'importance de la responsabilité des données dans la construction de la confiance entre les entreprises et leurs clients.

2.1.2 Alignement des valeurs

L'alignement des valeurs fait référence à la capacité des systèmes automatisés à prendre des décisions qui reflètent les valeurs et les normes éthiques de la société. Cela implique que les algorithmes d'IA doivent être conçus pour éviter les biais et les discriminations, tout en respectant les principes d'équité et d'inclusion. Par exemple, un système de traitement automatisé des données utilisé dans le domaine de la santé doit être capable de prendre des décisions qui respectent les droits des patients et les normes médicales éthiques.

Cependant, des cas réels montrent que les systèmes d'IA peuvent mal interpréter les données et produire des résultats biaisés, ce qui pourrait avoir des conséquences dramatiques. Par exemple, dans le cas de la conduite autonome, il est essentiel de comprendre le raisonnement qui sous-tend la décision d'un système d'IA de prendre une mesure particulière pour garantir la sécurité des utilisateurs [6]. Ce type de problème met en évidence la nécessité de concevoir des systèmes qui alignent leurs décisions sur des valeurs éthiques claires.

2.1.3 Redevabilité algorithmique

La redevabilité algorithmique concerne la capacité à expliquer et à justifier les décisions prises par les systèmes automatisés. Cela inclut la transparence des processus décisionnels et la capacité à identifier les responsables en cas de résultats erronés ou préjudiciables. Par exemple, si un algorithme de recrutement rejette injustement des candidats qualifiés, il doit être possible de retracer les

raisons de cette décision et de corriger le système.

La redevabilité algorithmique est particulièrement importante dans des domaines sensibles comme la justice, la santé, ou les services financiers, où les décisions automatisées peuvent avoir un impact significatif sur la vie des individus. Selon l'étude de l'IBM Institute for Business Value, deux tiers des dirigeants techniques considèrent la redevabilité algorithmique comme une priorité majeure, reflétant la demande croissante de transparence et d'explicabilité dans les systèmes d'IA [7].

2.1.4 Inclusion et équité

L'inclusion et l'équité sont des principes fondamentaux dans le traitement automatisé des données. Les systèmes d'IA doivent être conçus pour éviter les discriminations et garantir que tous les individus, indépendamment de leur genre, origine ethnique ou statut socio-économique, bénéficient de décisions justes et équitables. Cela nécessite une attention particulière à la diversité des données utilisées pour entraîner les algorithmes, ainsi qu'à la détection et à la correction des biais.

Par exemple, Amazon a dû abandonner un système de recrutement automatisé qui favorisait les candidats masculins en raison de biais présents dans les données historiques [5]. Cet exemple illustre l'importance de l'inclusion et de l'équité dans la conception des systèmes automatisés.

2.2 Normes éthiques dans l'IA

Les normes éthiques dans l'intelligence artificielle (IA) sont essentielles pour garantir que les systèmes d'IA soient développés et utilisés de manière responsable, équitable et respectueuse des droits humains. Les principes éthiques de l'IA incluent la responsabilité des données, l'alignement des valeurs, la redevabilité algorithmique et l'inclusion [8]. Ces principes visent à minimiser les

biais, à protéger la vie privée, et à assurer que les décisions prises par les systèmes d'IA soient transparentes et justifiables. Par exemple, une étude de l'IBM Institute for Business Value souligne que 81 pour cent des consommateurs sont préoccupés par l'utilisation de leurs données [7], ce qui renforce la nécessité de normes éthiques strictes pour maintenir la confiance du public. De plus, il est mis en avant cinq principes clés pour une IA éthique : la bienveillance, la non-malfaisance, l'indépendance, la justice, et l'interprétabilité. Ces principes sont alignés avec les recommandations de l'UNESCO et de l'IEEE, qui insistent sur la nécessité de créer des systèmes d'IA qui profitent à l'humanité tout en respectant les droits fondamentaux [9]. Enfin, l'étude de l'IBM Institute for Business Value mentionne que des pays comme la Jordanie intègrent désormais des programmes éducatifs sur l'éthique de l'IA, ce qui montre une prise de conscience croissante de l'importance de ces normes dans le développement technologique.[7].

2.3 TRANSPARENCE DANS LES OUTILS AUTOMATISES

La transparence dans les outils automatisés est un pilier central pour garantir la confiance et la compréhension des systèmes d'IA. La transparence implique que les organisations doivent être claires sur la source des données, le fonctionnement des algorithmes et les objectifs des systèmes d'IA [8]. Cela permet aux utilisateurs de comprendre comment les décisions sont prises et de détecter d'éventuelles erreurs ou biais. De plus, la transparence est cruciale pour éviter les discriminations et les injustices. Les développeurs d'IA doivent fournir des explications claires sur les processus décisionnels, en particulier dans des domaines sensibles comme la santé, la justice et l'emploi. Enfin, la transparence est également liée à la protection des données, car les utilisateurs doivent savoir comment leurs informations sont collectées, utilisées et protégées.

Malgré son importance, la transparence dans les systèmes d'IA pose plusieurs défis. L'un des principaux défis est la complexité des algorithmes d'apprentis-

sage profond, qui fonctionnent souvent comme des ”boîtes noires”[8]. Cela signifie que même les développeurs peuvent avoir du mal à expliquer comment les décisions sont prises, ce qui rend la transparence difficile à mettre en œuvre. Les attentes des consommateurs en matière de transparence sont élevées, mais les entreprises peuvent hésiter à divulguer des informations sensibles sur leurs algorithmes pour des raisons de propriété intellectuelle ou de concurrence. De plus, il existe de nombreux défis liés à la transparence, en particulier dans les pays en développement. Le manque de ressources techniques et de formation peut limiter la capacité des organisations à rendre leurs systèmes d'IA transparents. Par exemple, les algorithmes de détection de la couleur de peau et du genre peuvent avoir des taux d'erreur élevés (jusqu'à 16 pour cent pour les visages noirs), ce qui souligne la difficulté de garantir une transparence totale dans des systèmes complexes. Enfin, un autre défi est la nécessité de trouver un équilibre entre la transparence, la protection des données et la sécurité, car une trop grande transparence pourrait exposer des informations sensibles sur les individus ou les rendre vulnérables aux attaques ou aux vols.[10].

Cette section nous a permis de démontrer que l'automatisation du pré-traitement des données permet de gagner en efficacité, mais elle soulève des défis majeurs en matière d'éthique et de transparence. Les solutions actuelles, bien qu'avancées, nécessitent encore des ajustements pour répondre aux exigences d'équité et de transparence imposées par les législations en vigueur. Des normes éthiques, comme celles de l'UNESCO, encadrent l'utilisation de l'IA et insistent sur l'importance de préserver les droits de l'homme dans un monde dans lequel les décisions sont de plus en plus influencées par les données et les algorithmes.

Deuxième partie

MATERIELS
ET
METHODES

Chapitre 3

MATERIELS

Cette section présente en détail l'environnement technique, les outils et les méthodes utilisés pour développer FairAutoClean, l'outil de prétraitement automatique et éthique des données.

3.1 Analyse des besoins fonctionnels et techniques

3.1.1 Analyse des besoins fonctionnels

Le développement de FairAutoClean répond à plusieurs besoins fonctionnels essentiels :

Prétraitement automatique et efficient des données

L'objectif de cette étape est de préparer les données pour l'analyse en les nettoyant et en les transformant de manière à les rendre exploitables. Cela inclut la suppression des doublons, la gestion des valeurs manquantes, la détection et le traitement des valeurs aberrantes, la conversion des colonnes de date, l'encodage des colonnes catégorielles, et la normalisation des colonnes numériques.

Suppression des doublons :

— Identifier et éliminer les enregistrements redondants pour garantir l'unicité des données.

— Assurer que chaque ligne du jeu de données représente une entrée unique, ce qui améliore la qualité des analyses ultérieures.

Gestion des valeurs manquantes :

- **Imputation par régression** : utiliser des modèles de régression pour prédire et remplacer les valeurs manquantes dans les colonnes numériques.
- **Imputation KNN** : utiliser l'algorithme des k-plus proches voisins pour imputer les valeurs manquantes en tenant compte des relations entre les variables.
- **Suppression intelligente** : supprimer les lignes contenant un nombre excessif de valeurs manquantes tout en préservant la structure globale du jeu de données.

Gestion des valeurs aberrantes :

- **Détection par IQR** : utiliser la méthode de l'intervalle interquartile (IQR) pour identifier les valeurs extrêmes dans les données.
- **Traitement par winsorisation** : limiter l'impact des valeurs aberrantes en les remplaçant par des valeurs situées aux bornes de l'intervalle acceptable.

Détection et conversion des colonnes de date :

- **Identification automatique** : détecter les formats de date dans les colonnes et les convertir en un format standard.
- **Standardisation** : uniformiser les formats de date pour faciliter les analyses temporelles.

Encodage des colonnes catégorielles :

- **Label Encoding** : convertir les variables catégorielles ordinales en valeurs numériques pour les modèles d'apprentissage automatique.
- **One-Hot Encoding** : créer des colonnes binaires pour chaque catégorie dans les variables nominales, permettant une représentation sans ordre implicite.

Normalisation des colonnes numériques :

- **Standardisation** : transformer les données pour avoir une moyenne de 0 et un écart-type de 1, ce qui est utile pour les algorithmes sensibles à l'échelle.
- **Min-Max Scaling** : redimensionner les données dans un intervalle spécifique (par exemple, [0, 1]) pour les algorithmes basés sur des distances.
- **Robust Scaling** : utiliser des statistiques robustes (médiane et intervalle interquartile) pour normaliser les données en présence de valeurs aberrantes.

Réduction de dimension des données

La réduction de dimension vise à simplifier les données en réduisant le nombre de variables tout en conservant l'information essentielle. Cela améliore l'efficacité des modèles d'apprentissage automatique et réduit le temps de calcul.

PCA (Principal Component Analysis) :

- **Préservation de la variance** : réduire la dimensionnalité en projetant les données sur des axes orthogonaux qui maximisent la variance.
- **Configuration flexible** : permettre à l'utilisateur de choisir le nombre de composantes principales ou le pourcentage de variance à conserver.

Autoencoders :

- **Réduction non-linéaire** : utiliser des réseaux de neurones pour apprendre une représentation compressée des données, adaptée aux structures complexes.
- **Architecture adaptative** : adapter la profondeur et la complexité du modèle en fonction de la nature des données.

Détection automatique des colonnes sensibles

La détection des colonnes sensibles est cruciale pour garantir la conformité aux régulations éthiques et légales.

Détection algorithmique :

— **Listes de mots-clés** : utiliser des listes prédéfinies pour identifier les colonnes susceptibles de contenir des informations sensibles (par exemple, ”nom”, ”adresse”, ”salaire”).

— **Analyse des patterns** : examiner les noms de colonnes pour détecter des motifs courants associés à des données sensibles.

Détection par IA :

— **Grands modèles de langage (LLM)** : utiliser des modèles de langage avancés pour analyser le contexte des colonnes et évaluer leur sensibilité.

— **Évaluation de la sensibilité** : attribuer un score de sensibilité sur une échelle de 0 à 10 pour chaque colonne, en fonction du niveau de sensibilité potentiel.

Détection des biais algorithmiques

Dans certains cas de figures, il peut arriver que les données aient déjà été traitées au préalable, ou qu'un traitement supplémentaire a été prévu d'être fait. Dans ces cas, il est utile de pouvoir analyser le code qui fera ce prétraitement, afin de détecter les potentiels erreurs qu'il pourrait introduire.

Analyse syntaxique du code :

— **Identification des patterns discriminatoires** : examiner le code source afin d'identifier des schémas ou opérations susceptibles d'introduire des biais, notamment ceux impliquant des filtres ou des sélections basées sur des caractéristiques sensibles (genre, race, âge, etc.).

— **Vérification des opérations de filtrage** : analyser les algorithmes de filtrage et de sélection pour vérifier qu'ils ne favorisent ni ne désavantagent injustement certains groupes d'individus.

Analyse par IA :

— **Utilisation de grands modèles de langage (LLM)** : employer des modèles de langage avancés pour évaluer les implications éthiques du code, en identifiant les zones à risque et en suggérant des améliorations.

— **Génération de recommandations** : produire des recommandations concrètes pour modifier le code afin de réduire les biais et d'améliorer l'équité des décisions algorithmiques.

Visualisation et audit

La visualisation et l'audit des données et des processus sont cruciaux pour assurer la transparence, la redevabilité et la qualité des systèmes d'IA. Ces étapes permettent de comprendre les données, de détecter les anomalies et de documenter les opérations effectuées.

Exploratory Data Analysis (EDA) :

— **Génération de rapports de profiling** : créer des rapports détaillés qui résument les caractéristiques des données, telles que les distributions, les valeurs manquantes et les corrélations entre variables.

— **Visualisations statistiques** : produire des graphiques et des diagrammes (histogrammes, boîtes à moustaches, etc.) pour visualiser les distributions des données et identifier les tendances ou les anomalies.

Traçabilité :

— **Journalisation des opérations** : enregistrer de manière détaillée toutes les opérations effectuées sur les données, y compris les transformations, les nettoyages, et les analyses, pour assurer une transparence totale.

— **Rapport d'audit complet** : générer un rapport d'audit qui inclut des métriques de performance, des indicateurs de qualité des données, et des évaluations des biais, permettant une vérification externe et une conformité aux normes éthiques et réglementaires.

3.1.2 Analyse des besoins techniques

Pour développer un outil automatisé de prétraitement éthique et transparent des données, plusieurs besoins techniques doivent être identifiés et satisfaits. Ces besoins incluent le choix d'un langage de programmation adapté, la sélection d'un environnement de développement efficace, et l'utilisation de bibliothèques et d'outils spécialisés pour la gestion des données, l'apprentissage automatique, et la visualisation. Voici une liste des outils et technologies qui seront utilisés pour répondre à ces besoins :

- **Langage de programmation** : python
- **Environnement de développement** : VSCode, Git, et les environnements virtuels
- **Bibliothèques de data science** : Pandas, NumPy, Scikit-learn
- **Outils de visualisation** : Matplotlib, Seaborn
- **Gestion des dépendances** : Pip, Conda
- **Manipulation des LLM** : PydanticAI
- **Détection et gestion des biais** : AI Fairness 360

Langage de programmation

Python a été choisi comme langage de programmation principal pour ce projet en raison de ses nombreux avantages dans le domaine de la data science et de l'intelligence artificielle.

- **Richesse en bibliothèques** : python dispose d'un écosystème riche en bibliothèques spécialisées pour le traitement des données (Pandas, NumPy), l'apprentissage automatique (Scikit-learn, TensorFlow), et la visualisation (Matplotlib, Seaborn). Ces bibliothèques permettent de développer des solutions robustes et efficaces.
- **Communauté active** : python bénéficie d'une communauté de développeurs très active, ce qui facilite l'accès à des ressources, des tutoriels, et des solutions aux problèmes courants.

— **Facilité d'intégration** : python s'intègre facilement avec d'autres outils et technologies, ce qui en fait un choix idéal pour des projets nécessitant une interopérabilité avec des systèmes existants.

Figure 3.1 – Logo de Python, le langage de programmation choisi pour ce projet.

Environnement de développement

Un environnement de développement moderne et efficace est essentiel pour assurer la productivité et la qualité du code. Les outils suivants seront utilisés :

VSCode :

— Un éditeur de code léger mais puissant, avec un support natif pour Python.

— Offre des fonctionnalités avancées comme le débogage, la complétion de code, et l'intégration avec Git.

— Extensible via des extensions pour répondre à des besoins spécifiques (par exemple, des extensions pour la data science ou l'IA).

Figure 3.2 – Logo de VSCode, l'éditeur de code utilisé pour ce projet.

Git :

— Un système de contrôle de version distribué pour suivre les modifications du code et collaborer efficacement.

— Permet de gérer les branches, les fusions, et les versions du projet de manière structurée.

— Intégré avec des plateformes comme GitHub ou GitLab pour le stockage et le partage du code.

Figure 3.3 – Logo de Git, le système de contrôle de version utilisé pour ce projet.

Environnement virtuel :

— Utilisation d'environnements virtuels (par exemple, avec 'venv' ou 'conda') pour isoler les dépendances du projet.

— Permet de gérer les versions des bibliothèques et d'éviter les conflits entre projets.

— Facilite la reproductibilité des résultats en garantissant que tous les membres de l'équipe utilisent les mêmes versions des outils.

Manipulation des LLM avec PydanticAI

Pour intégrer des modèles de langage (LLM) dans le projet, la bibliothèque PydanticAI a été choisie. Cette bibliothèque offre des fonctionnalités avancées pour interagir avec des LLM tout en garantissant une gestion structurée des données et des réponses.

— **Structuration des données** : PydanticAI permet de définir des schémas de données robustes pour les entrées et les sorties des LLM, ce qui facilite la validation et la manipulation des données.

— **Intégration avec les LLM** : la bibliothèque supporte l'intégration avec des modèles de langage populaires comme GPT, BERT, et d'autres, permettant des interactions fluides et efficaces.

— **Gestion des erreurs** : PydanticAI inclut des mécanismes de gestion des erreurs pour traiter les réponses inattendues ou invalides des LLM, améliorant ainsi la robustesse du système.

Figure 3.4 – Logo de PydanticAI

Détection et gestion des biais avec AI Fairness 360

Pour garantir que les systèmes d'IA soient équitables et non discriminatoires, la bibliothèque AI Fairness 360 (AIF360) a été sélectionnée. Cette bibliothèque open-source, développée par IBM, fournit des outils pour détecter, mesurer, et atténuer les biais dans les modèles d'IA.

— **Détection des biais** : AI Fairness 360 inclut des algorithmes pour identifier les biais dans les données et les modèles, en analysant les disparités entre différents groupes (par exemple, genre, race, âge).

— **Atténuation des biais** : la bibliothèque propose des techniques pour réduire les biais, telles que la réévaluation des poids des données, l'ajustement des seuils de décision, et l'utilisation de méthodes de rééchantillonnage.

— **Métriques d'équité** : AI Fairness 360 fournit des métriques pour évaluer l'équité des modèles, comme l'égalité des chances, la parité démographique, et l'impact disparate.

Figure 3.5 – Logo de Ai Fairness 360.

3.1.3 Environnement technique

Pour garantir une exécution fluide et efficace des tâches de développement et d'analyse, il est essentiel de disposer d'un environnement technique adapté. Cette section décrit la configuration matérielle utilisée pour ce projet, ainsi que les spécifications techniques qui permettent de supporter les charges de travail liées au prétraitement des données, à l'apprentissage automatique, et à la gestion des modèles de langage.

Configuration matérielle

- **Ordinateur portable** : HP Laptop 14s-dq2395nia
- **Processeur** : 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz (4 cœurs, 8 threads)
- **Mémoire RAM** : 16,0 Go (DDR4)
- **Système d'exploitation** : Windows 11 (64 bits)

Chapitre 4

METHODES

4.1 Architecture logicielle

L'architecture de FairAutoClean a été conçue avec une approche modulaire pour garantir une maintenabilité optimale et une extensibilité future. Cette structure permet de séparer les responsabilités fonctionnelles en modules distincts, facilitant ainsi la gestion du code, les tests unitaires et l'ajout de nouvelles fonctionnalités. Chaque module est conçu pour être indépendant et réutilisable, ce qui améliore la flexibilité du système et permet une intégration aisée avec d'autres outils ou frameworks. L'architecture repose sur une interface unifiée, gérée par le module `interface.py`, qui sert de point d'entrée pour toutes les interactions avec le système.

Structure générale

— **Architecture en couches avec séparation claire des responsabilités** : L'architecture de FairAutoClean est organisée en plusieurs couches logiques, chacune ayant une responsabilité spécifique. Cette séparation permet une meilleure gestion des dépendances et une plus grande clarté dans l'organisation du code. Les couches principales incluent la gestion des données, le traitement des données, l'analyse des biais et la génération de rapports.

— **Modules indépendants et réutilisables** : Chaque fonctionnalité du système est encapsulée dans un module distinct. Par exemple, le module `DataProcessor`

est responsable du chargement et du prétraitement des données, tandis que le module `FairnessAnalyzer` se concentre sur la détection et la mitigation des biais. Cette modularité permet de réutiliser les composants dans d'autres projets ou de les mettre à jour indépendamment sans affecter l'ensemble du système.

— **Interface unifiée via le module `interface.py`** : Le module `interface.py` sert de point d'entrée unique pour toutes les interactions avec FairAutoClean. Il gère la communication entre les différents modules et expose une API simple et cohérente pour les utilisateurs. Cette interface unifiée simplifie l'intégration de FairAutoClean dans des workflows existants et permet une configuration facile via des fichiers JavaScript Object Notation (JSON) ou des paramètres en ligne de commande.

Composants principaux

— **AutoClean** : Cette classe est le cœur du système. Elle orchestre le pipeline de nettoyage des données en coordonnant les différentes étapes du prétraitement, telles que la gestion des valeurs manquantes, la détection des valeurs aberrantes et la normalisation des données. Elle garantit que chaque étape est exécutée dans le bon ordre et que les données sont correctement transformées avant d'être analysées.

— **DataProcessor** : Ce module est responsable du chargement des données brutes et de leur préparation pour le traitement. Il inclut des fonctionnalités pour la validation des données, la conversion des formats, et la gestion des erreurs. Le `DataProcessor` assure que les données sont dans un état exploitable avant d'être transmises aux autres modules.

— **FairnessAnalyzer** : Ce composant est dédié à la détection et à la mitigation des biais dans les données. Il utilise des algorithmes avancés pour identifier les disparités entre différents groupes (par exemple, genre, race, âge) et propose des corrections pour garantir l'équité des résultats. Le `FairnessAnalyzer` est essentiel pour respecter les normes éthiques et légales en matière de traitement des données.

— **AICodeAnalyzer** : Ce module analyse le code source utilisé pour le prétraitement des données afin de détecter d'éventuels biais algorithmiques. Il examine les opérations de filtrage, de transformation, et de sélection pour s'assurer qu'elles ne favorisent pas ou ne défavorisent pas injustement certains groupes. Le `AICodeAnalyzer` génère également des recommandations pour améliorer l'équité du code.

— **ReportGenerator** : Ce composant est chargé de générer des rapports détaillés sur les opérations effectuées par FairAutoClean. Il produit des métriques de performance, des indicateurs de qualité des données, et des évaluations des biais. Ces rapports sont essentiels pour assurer la transparence et la traçabilité du processus de prétraitement, permettant aux utilisateurs de comprendre et d'auditer les actions entreprises par le système.

4.2 Flux de données et architecture système

Le système FairAutoClean est organisé en plusieurs couches logiques, chacune ayant un rôle spécifique dans le processus de prétraitement des données. Cette architecture en couches permet une séparation claire des responsabilités, une meilleure gestion des dépendances et une extensibilité accrue. Voici une description de chaque couche et de ses composants :

1. **Couche d'entrée** : Cette couche est responsable de la réception et de la préparation des données et des configurations nécessaires au fonctionnement du système. Elle comprend deux éléments principaux :
   — **Données d'entrée** : Les données brutes sont chargées dans le système en format CSV ou Excel, qui sont des formats de données largement utilisés dans le domaine des data sciences en particulier et dans tous les secteurs en général
   — **Configuration JSON** : Un fichier de configuration au format JSON est utilisé pour paramétrer le comportement du système. Ce fichier contient des informations telles que les méthodes de prétraitement à appliquer, les seuils pour la détection des valeurs aberrantes, les paramètres de réduction de dimensionnalité, etc. La configuration JSON

permet une personnalisation fine du processus de prétraitement sans nécessiter de modifications du code source.

2. **Couche d'interface** : Cette couche fournit des interfaces pour interagir avec le système, que ce soit via une interface en ligne de commande (CLI) ou une API Python. Elle comprend :
   - **Interface CLI** : Une interface en ligne de commande permet aux utilisateurs de configurer et d'exécuter FairAutoClean avec des paramètres personnalisés. Elle est idéale pour les utilisateurs techniques ou pour l'intégration dans des scripts automatisés.
   - **Package Python** : FairAutoClean est également disponible sous forme de package Python, permettant une intégration facile dans des projets existants. Les utilisateurs peuvent importer les modules et les fonctions directement dans leur code.
   - **Gestion de la configuration** : Ce composant permet aux utilisateurs de configurer le système via des fichiers JSON ou des paramètres en ligne de commande. Il garantit une flexibilité maximale dans la personnalisation du processus de prétraitement.

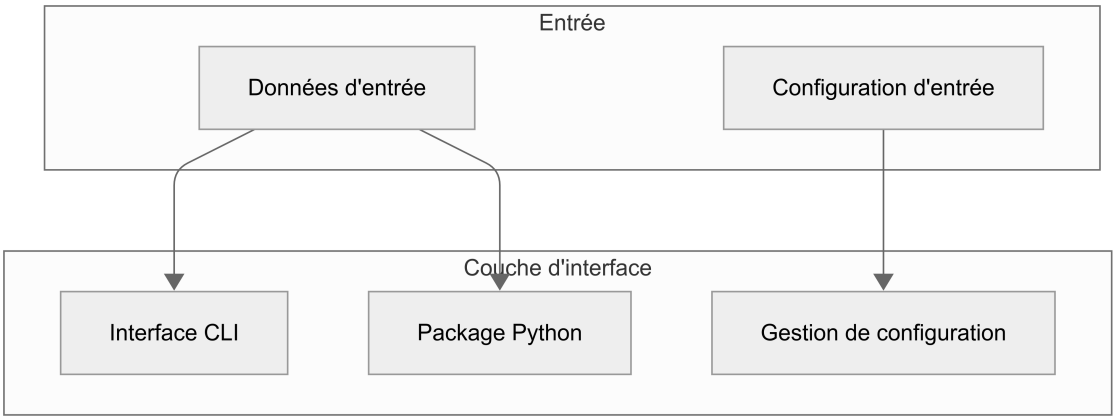

Figure 4.1 – Représentation des composants de la couche d'entrée des données

3. **Couche de traitement principal** : Cette couche est le cœur du système, où les données sont nettoyées, transformées et préparées pour l'analyse. Elle comprend plusieurs modules clés :
   - **Processeur de données** : Ce module charge les données brutes, valide leur intégrité, et les prépare pour le traitement. Il gère également les erreurs de format ou les données manquantes en appliquant des stratégies prédéfinies (par exemple, l'imputation ou la suppression des valeurs manquantes).
   - **Nettoyeur de données** : Ce module effectue des opérations de nettoyage telles que la suppression des doublons, la gestion des valeurs aberrantes, et la normalisation des données. Il garantit que les données sont propres et cohérentes avant d'être transmises aux étapes suivantes.
   - **Normalisateur** : Ce composant applique des transformations pour standardiser les données, comme la normalisation des valeurs numériques ou l'encodage des variables catégorielles. Ces transformations sont essentielles pour garantir que les données sont adaptées aux algorithmes d'apprentissage automatique.
   - **Réduction de dimensionnalité** : Ce module réduit la complexité des données en supprimant les variables redondantes ou peu informatives. Des techniques telles que l'Analyse en Composantes Principales (PCA) ou les autoencodeurs sont utilisées pour conserver l'information essentielle tout en réduisant le temps de calcul.
   - **Ingénierie des caractéristiques** : Ce module crée de nouvelles variables ou caractéristiques à partir des données existantes pour améliorer la performance des modèles d'apprentissage automatique. Il peut inclure des opérations telles que la création d'interactions entre variables ou l'extraction de caractéristiques temporelles.

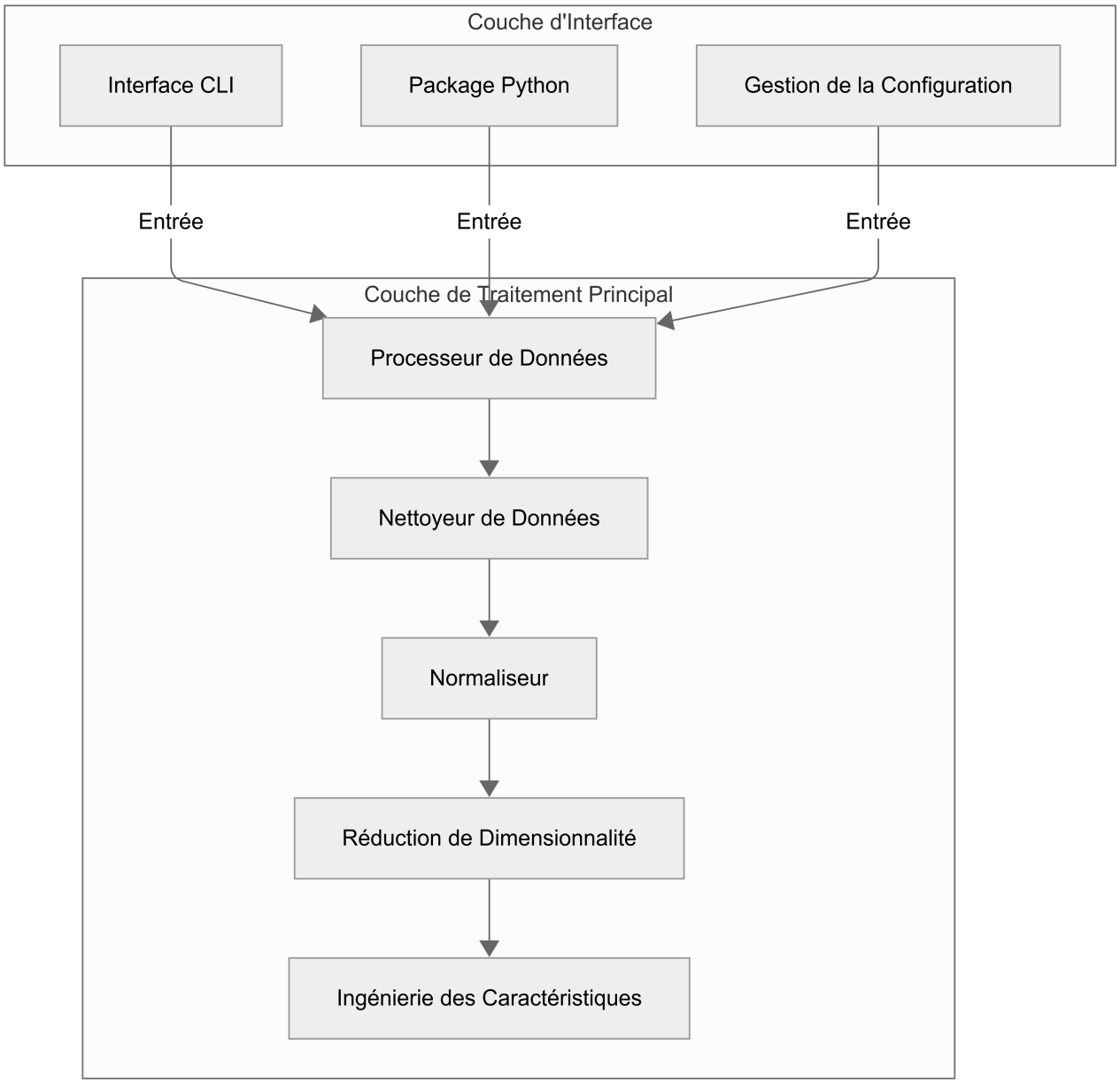

Figure 4.2 – Représentation des composants de la couche de traitement principal

4. **Couche d'équité et d'analyse** : Cette couche est dédiée à la détection et à la mitigation des biais, ainsi qu'à l'analyse éthique des données et du code. Elle comprend les composants suivants :
   — **Analyseur d'équité** : Ce module utilise des algorithmes pour identifier les biais dans les données, tels que les disparités entre groupes démographiques. Il propose des corrections pour garantir que les données sont équitables et conformes aux normes éthiques.
   — **Analyseur de code** : Ce composant examine le code source utilisé

pour le prétraitement des données afin de détecter d'éventuels biais algorithmiques. Il génère des rapports détaillés sur les risques éthiques et propose des recommandations pour améliorer l'équité du code.

— **Système d'audit** : Ce module enregistre toutes les opérations effectuées par le système, y compris les transformations des données, les détections de biais, et les corrections appliquées. Il garantit une transparence totale et permet aux utilisateurs de vérifier et d'auditer le processus de prétraitement.

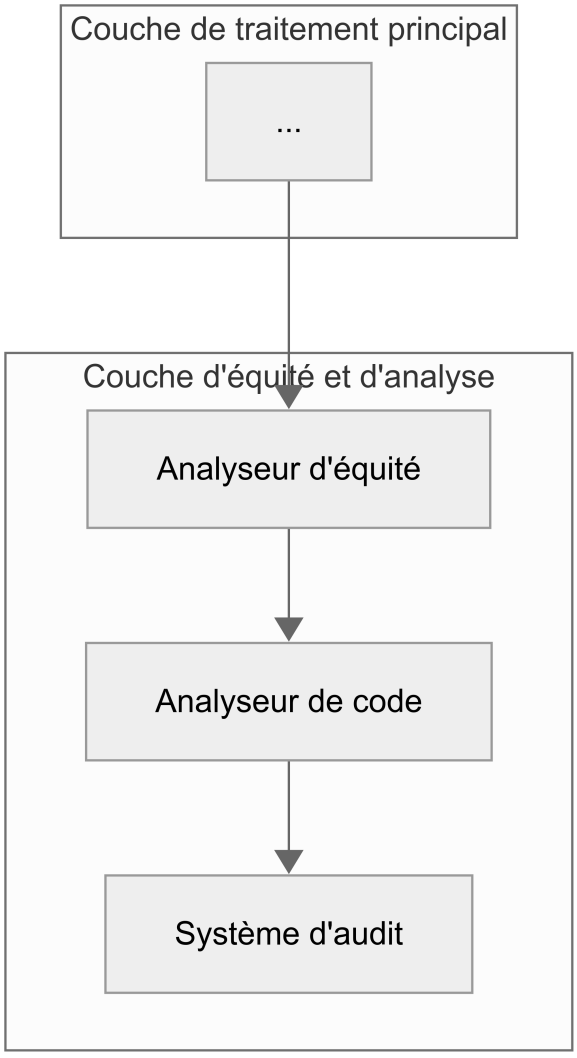

Figure 4.3 – Représentation des composants de la couche d'équité et d'analyse

5. **Système d'agent IA** : Cette couche intègre des modèles de langage (LLMs) et des agents intelligents pour améliorer les fonctionnalités du système. Elle

comprend :

— **Intégration LLM** : Des modèles de langage avancés, tels que GPT, sont utilisés pour analyser le contexte des données et générer des recommandations éthiques. Ces modèles permettent également de détecter des biais subtils qui pourraient échapper aux algorithmes traditionnels.

— **Gestionnaire de prompts système** : Ce composant génère des prompts adaptés pour interagir avec les LLMs, garantissant des réponses précises et pertinentes. Il est essentiel pour maximiser l'efficacité des interactions avec les modèles de langage.

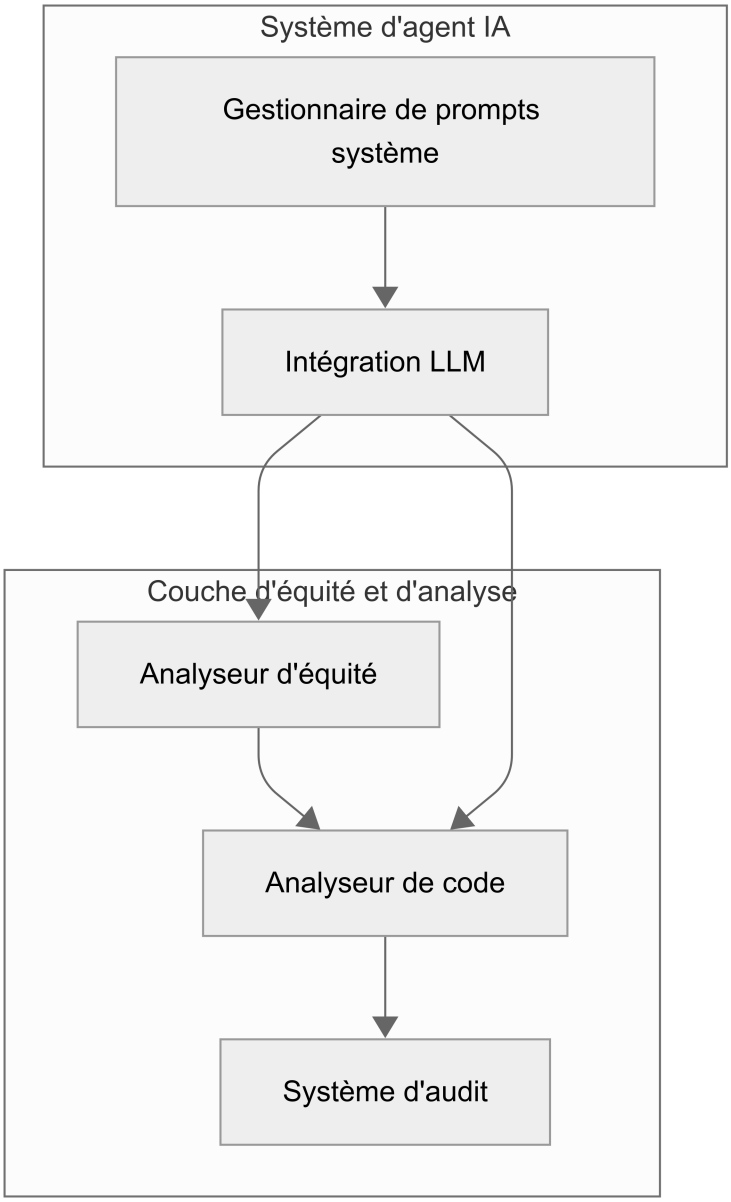

Figure 4.4 – Représentation des composants du système d'agent IA

6. **Sortie et rapports** : Cette couche est responsable de la génération des résultats finaux et des rapports détaillés. Elle comprend :
   - **Générateur de rapports** : Ce module produit des rapports complets sur les opérations effectuées, les biais détectés, et les corrections appliquées. Ces rapports sont essentiels pour assurer la transparence et la traçabilité du processus.
   - **Générateur de profils** : Ce composant crée des profils détaillés des données, incluant des statistiques descriptives, des visualisations, et

des indicateurs de qualité. Ces profils aident les utilisateurs à comprendre les caractéristiques des données avant et après le prétraitement.

— **Générateur de logs d'audit** : Ce module enregistre toutes les actions entreprises par le système, créant une piste d'audit complète pour chaque exécution. Cette fonctionnalité est cruciale pour répondre aux exigences réglementaires et éthiques.

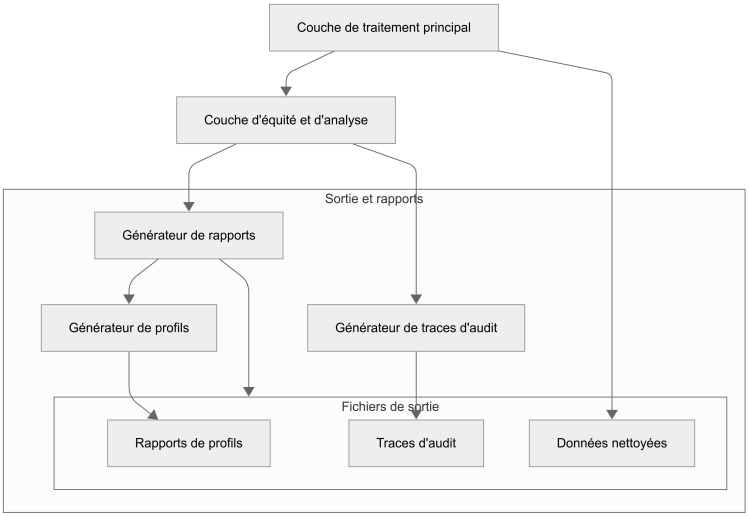

Figure 4.5 – Représentation des composants de la partie sortie et rapports

4.3 Configuration et utilisation de l'outil

FairAutoCleaner est un outil innovant de prétraitement automatisé des données, conçu pour garantir à la fois l'efficacité technique et l'équité éthique dans le traitement des données. Ce chapitre présente en détail les aspects de configuration et d'utilisation de l'outil.

Configuration de l'Outils

4.3.0.1 Structure du Fichier de Configuration

Le fichier JSON de configuration se structure en trois parties principales :

4.3.0.2 Métadonnées du Dataset

```
"dataset": {
    "title": "Jeu de données bancaires",
    "description": "Données clients pour prédiction d'attrition",
    "target": "churn",
    "sensitive_features": ["genre", "age"]
}
```

Figure 4.6 – Exemple de configuration pour les métadonnées du dataset

4.3.0.3 Paramètres de Prétraitement

```
"preprocessing": {
    "normalization": {
        "enabled": true,
        "method": "standard",
        "exclude_features": ["id_client"]
    },
    "dim_reduction": {
        "enabled": true,
        "method": "pca",
        "target_explained_variance": 0.95
    }
}
```

Figure 4.7 – Exemple de configuration pour les paramètres de prétraitement

4.3.0.4 Analyse de Code

Cette section optionnelle permet d'analyser les scripts de prétraitement pour détecter d'éventuels biais algorithmiques. Elle offre :

— La possibilité de spécifier les répertoires à analyser

— Le choix entre analyse syntaxique ou IA

— Un niveau de sensibilité configurable (1-10)

— Une traçabilité complète des analyses effectuées

— Des recommandations concrètes pour corriger les biais détectés

— La génération automatique de rapports détaillés

```
"code_analysis": {
    "paths": ["scripts/preprocessing"],
    "type": "ai",
    "sensitivity_level": 7,
    "report_level": "detailed",
    "exclude_patterns": ["test_*.py"],
    "max_file_size": 1048576
}
```

Figure 4.8 – Exemple de configuration pour l'analyse de code

Processus d'Utilisation

4.3.0.5 Workflow Principal

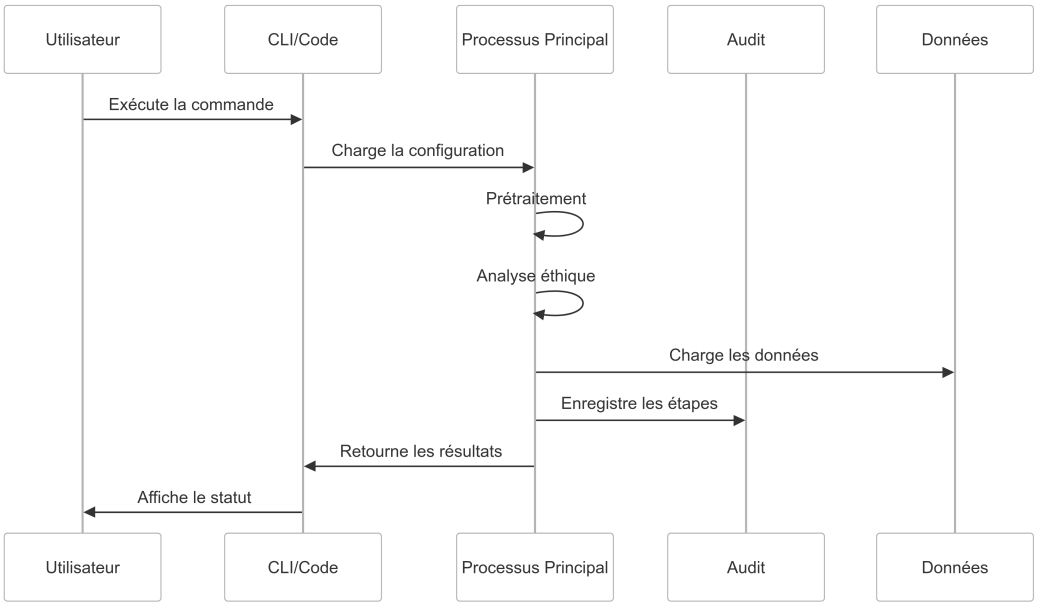

Figure 4.9 – Workflow principal de FairAutoCleaner

Le workflow principal de FairAutoCleaner suit une séquence d'interactions bien définie entre l'utilisateur et les différents composants du système. L'utilisateur initie le processus via l'interface en ligne de commande (CLI) qui or-

chestre l'exécution des différentes étapes. Le processus principal charge d'abord la configuration et les données, puis effectue l'analyse éthique et le prétraitement tout en enregistrant chaque étape dans le système d'audit. Enfin, les résultats sont retournés à l'utilisateur via l'interface CLI.

1. Initialisation de l'audit
2. Chargement des données
3. Analyse éthique automatique
4. Prétraitement des données
5. Génération des rapports

4.3.0.6 Interface en Ligne de Commande

```
python -m FairAutoCleaner \
    --config configuration.json \
    --dataset donnees.csv \
    --output resultats/
```

Figure 4.10 – Interface en Ligne de Commande

4.3.0.7 API Python

```
from FairAutoCleaner import process_dataset
resultats = process_dataset(
    config_path="config.json",
    dataset_path="data.csv",
    output_path="output/"
)
```

Figure 4.11 – API Python

Résultats et Rapports

4.3.0.8 Structure des fichiers en sortie

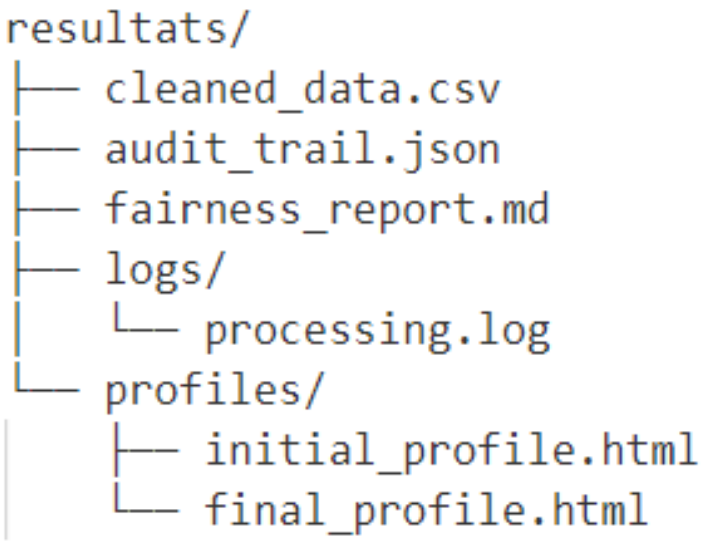

Figure 4.12 – Structure des fichiers en sortie

4.3.0.9 Journal d'Audit

Le fichier JSON d'audit contient :

- Timestamps des opérations
- Paramètres utilisés
- Métriques de performance
- Avertissements et erreurs
- Détails des transformations

Conclusion

FairAutoCleaner offre une solution complète pour le prétraitement éthique des données, combinant automatisation technique et vigilance éthique. Sa configuration modulaire et ses rapports détaillés en font un outil essentiel pour les projets de science des données responsables.

4.4 Implémentation de l'outil

4.4.1 Preprocessing automatique

Justification du pipeline de pretraitement choisi

L'ordre des étapes de prétraitement est crucial, car chaque opération peut affecter les données et influencer l'efficacité des étapes suivantes [11].

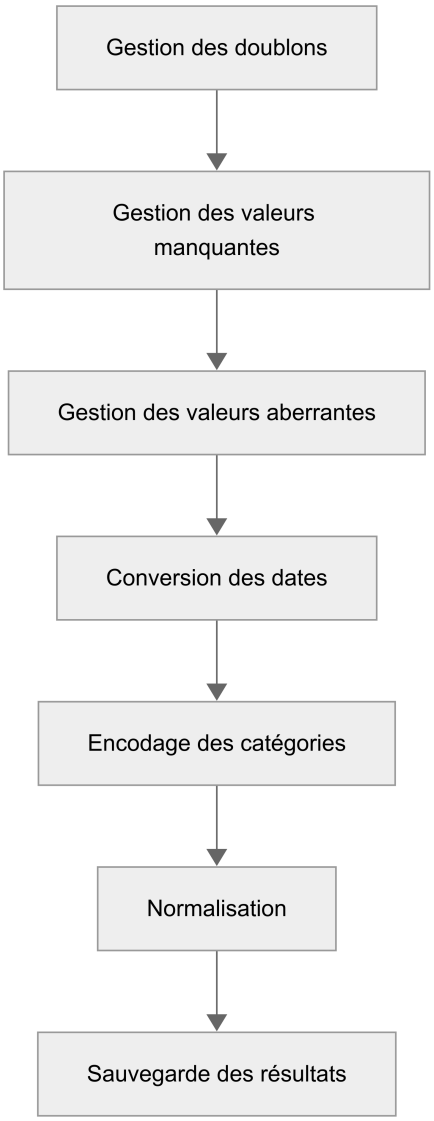

Figure 4.13 – Pipeline de nettoyage automatique des données

1. **Gestion des doublons** : Il est logique de commencer par éliminer les doublons. Les doublons peuvent introduire des biais et fausser les statistiques des données [1]. Les supprimer en premier assure que les étapes suivantes

sont appliquées à un ensemble de données unique.

2. **Gestion des valeurs manquantes** : Une fois les doublons éliminés, la gestion des valeurs manquantes est une étape essentielle [2]. Les valeurs manquantes peuvent perturber le fonctionnement de nombreux algorithmes d'apprentissage automatique. L'imputation (remplacement des valeurs manquantes par des estimations) est une technique courante. Il est préférable de traiter les valeurs manquantes avant de s'occuper des valeurs aberrantes ou de la normalisation, car ces dernières étapes pourraient être affectées par la présence de valeurs manquantes ou introduire des biais si elles sont traitées avant[2].
3. **Gestion des valeurs aberrantes** : Après avoir traité les valeurs manquantes, l'identification et la gestion des valeurs aberrantes sont importantes [12]. Les valeurs aberrantes peuvent avoir un impact significatif sur les statistiques des données et les performances des modèles. Des méthodes comme le Z-score peuvent être utilisées pour détecter les outliers [13]. Il est généralement conseillé de traiter les outliers avant la normalisation, car les outliers peuvent influencer les paramètres de normalisation (moyenne, écart type, min, max) et réduire l'efficacité de cette étape pour les données non aberrantes.
4. **Conversion des dates** : La conversion des colonnes de type datetime en formats numériques ou en caractéristiques exploitables (par exemple, l'année, le mois, le jour de la semaine) est une étape de préparation des caractéristiques [1]. Bien que n'ayant l'obligation de placer cette conversion à une étape précise, il est logique de la placer avant l'encodage catégoriel et la normalisation, car les informations temporelles extraites peuvent être de nature catégorielle ou numérique et nécessiter un traitement ultérieur.
5. **Encodage des valeurs catégorielles** : La plupart des modèles de deep learning fonctionnent mieux avec des données numériques et ne peuvent pas traiter directement les données catégorielles [4]. L'encodage des valeurs catégorielles en représentations numériques (par exemple, one-hot encoding, label encoding) est donc nécessaire. Cette étape doit précéder la

normalisation si la normalisation est appliquée uniquement aux caractéristiques numériques

6. **Normalisation** : Enfin, la normalisation ou la mise à l'échelle des caractéristiques numériques est souvent réalisée pour s'assurer que les caractéristiques ont une plage similaire. Cela peut améliorer la convergence de certains algorithmes et empêcher que les caractéristiques avec des valeurs plus importantes ne dominent les autres. Des techniques courantes incluent la standardisation et la normalisation min-max [11].

4.4.1.1 Gestion des Duplicatas dans le Prétraitement des Données

La gestion des duplicatas est une étape cruciale dans le prétraitement des données. Les données en double peuvent fausser les analyses statistiques et réduire la précision des modèles de machine learning. Dans cette section, nous allons explorer pourquoi il est important de gérer les duplicatas et comment cela s'inscrit dans notre processus global de prétraitement.

Importance de la Gestion des Duplicatas :

Les données en double peuvent provenir de diverses sources, notamment des erreurs lors de la saisie des données ou des fusions de bases de données. Ces duplicatas peuvent entraîner plusieurs problèmes :

— **Biais dans les analyses** : Les données en double peuvent amplifier certaines tendances ou caractéristiques, conduisant à des conclusions erronées [14].

— **Inexactitudes statistiques** : Les statistiques calculées sur des données contenant des duplicatas peuvent être faussées, ce qui affecte la fiabilité des résultats [15].

— **Impact sur les modèles de machine learning** : Les modèles apprennent à partir des données disponibles. Si ces données contiennent des duplicatas, les modèles risquent de surapprendre sur ces données répétitives, ce qui réduit leur capacité à généraliser correctement [16].

Étapes pour Gérer les Duplicatas :

1. **Identification des Duplicatas** : Utiliser des méthodes pour détecter les enregistrements en double. Cela peut être fait en comparant toutes les colonnes ou en sélectionnant des colonnes spécifiques pertinentes pour l'analyse [17].
2. **Suppression des Duplicatas** : Une fois les duplicatas identifiés, il est possible de les supprimer en gardant soit la première occurrence, soit la dernière, ou même en supprimant toutes les occurrences [18].

Implementation de la Fonction `drop_duplicates` dans Pandas :

Pandas, une bibliothèque Python très populaire pour la manipulation de données, propose une fonction puissante appelée `drop_duplicates` pour gérer les duplicatas dans les DataFrames.

Syntaxe et Paramètres :

La fonction `drop_duplicates` a la syntaxe suivante :

```
df.drop_duplicates(
    subset=None,
    keep='first',
    inplace=False,
    ignore_index=False
)
```

— **`subset`** : Spécifie les colonnes à considérer pour identifier les duplicatas. Si `None`, toutes les colonnes sont prises en compte [19].

— **`keep`** : Détermine quelle occurrence des duplicatas conserver. Les options sont `'first'`, `'last'`, ou `False` (supprime toutes les occurrences) [20].

— `inplace` : Si `True`, modifie le DataFrame original. Sinon, retourne un nouveau DataFrame [21].

— `ignore_index` : Si `True`, réinitialise l'index du DataFrame résultant [22].

En résumé, la gestion des duplicatas est essentielle pour garantir la qualité et la fiabilité des analyses de données. La fonction `drop_duplicates` de Pandas offre une solution efficace et flexible pour supprimer les enregistrements en double, permettant ainsi d'améliorer la précision des modèles et des analyses statistiques [23].

4.4.1.2 Gestion des valeurs aberrantes

La winsorization est une technique statistique robuste utilisée pour atténuer l'impact des valeurs aberrantes (**outliers**) tout en conservant l'intégrité structurelle des données. Contrairement à la suppression pure des outliers, cette méthode remplace les valeurs extrêmes par des valeurs situées à des percentiles prédéfinis, préservant ainsi la taille de l'échantillon et réduisant les distorsions induites par les anomalies.

Algorithme Technique de la Winsorization :
La méthode suit une séquence d'étapes précises pour traiter les données :

Définition des bornes :

— Sélectionner un seuil de winsorization, généralement exprimé en pourcentage (p. ex., 5% pour chaque extrémité)

— Calculer les **percentiles** correspondants :

  — **Percentile inférieur** (p. ex., 5^{e} percentile : $\boldsymbol{P}_{\text{inf}}$)

  — **Percentile supérieur** (p. ex., 95^{e} percentile : $\boldsymbol{P}_{\text{sup}}$) [24, 25]

Remplacement des Valeurs Extrêmes :

— Identifier toutes les valeurs $< \boldsymbol{P}_{\text{inf}}$ et les remplacer par $\boldsymbol{P}_{\text{inf}}$

— Identifier toutes les valeurs $> \boldsymbol{P}_{\text{sup}}$ et les remplacer par $\boldsymbol{P}_{\text{sup}}$

— Formellement, pour une observation x_i, la valeur winsorisée x'_i devient :

$$x'_i = \begin{cases} P_{\text{inf}} & \text{si } x_i < P_{\text{inf}}, \\ P_{\text{sup}} & \text{si } x_i > P_{\text{sup}}, \\ x_i & \text{sinon.} \end{cases}$$

Cette approche préserve la distribution globale tout en limitant l'influence des extrêmes [14, 26]

Recalcul des Statistiques :
Les statistiques descriptives (moyenne, écart-type) et les modèles ultérieurs sont calculés sur les données winsorisées, offrant une robustesse accrue [27, 28]

Aspects Techniques et Paramètres :

— **Choix des Percentiles** : La sélection des bornes dépend du contexte et du degré de conservatisme souhaité. Par exemple, une winsorization à 5% symétrique (des deux côtés) est courante, mais une approche asymétrique peut être adoptée si les outliers sont concentrés dans une seule queue [25, 14]

— **Gestion des Données Multivariées** : Dans les jeux de données multivariés, la winsorization peut être appliquée variable par variable pour éviter les interactions complexes entre dimensions [25, 14]

— **Intégration dans les Pipelines de Prétraitement** : La méthode s'intègre facilement dans des workflows automatisés via des bibliothèques comme SciPy (`winsorize()`) ou des implémentations personnalisées en Python, garantissant une reproductibilité et une scalabilité optimales [14, 26]

La winsorization est un compromis méthodologique entre la suppression totale des outliers et l'acceptation de leur influence disruptive. Son implémentation technique repose sur une définition rigoureuse des percentiles et une application systématique du remplacement des valeurs extrêmes.[24, 27, 26]

4.4.1.3 Conversion des dates

La conversion des dates est une étape cruciale dans le prétraitement des données, particulièrement lorsqu'il s'agit d'analyser des données temporelles. Les formats de dates peuvent varier considérablement selon les sources et les systèmes utilisés, ce qui rend leur standardisation indispensable pour une analyse efficace.

Importance de la Conversion des Dates :

Les formats de dates peuvent être très diversifiés, allant de formats simples comme "JJ/MM/AAAA" à des formats plus complexes incluant l'heure et le fuseau horaire. La conversion des dates est essentielle pour plusieurs raisons :

- **Standardisation** : Un format de date standardisé facilite la comparaison et l'analyse des données temporelles.
- **Calculs Temporels** : Pour effectuer des calculs temporels, comme la différence entre deux dates ou l'identification de périodes spécifiques, il est crucial d'avoir des dates au format approprié.
- **Intégration avec d'autres Données** : Les données temporelles sont souvent intégrées avec d'autres types de données. Une standardisation des dates facilite cette intégration.

Étapes pour la Conversion des Dates :

1. **Identification des Formats de Dates** : Reconnaître les différents formats de dates présents dans les données.
2. **Choix d'un Format Standard** : Sélectionner un format standard pour toutes les dates, par exemple "AAAA-MM-JJ" ou "AAAA-MM-JJ HH :MM :SS".
3. **Conversion des Dates** : Utiliser des outils ou des bibliothèques pour convertir toutes les dates vers le format choisi.

Utilisation de Pandas pour la Conversion des Dates :

Pandas, une bibliothèque Python très utilisée pour la manipulation de données

[19], propose des outils puissants pour convertir et manipuler les dates. La gestion des dates en Python s'appuie également sur le module standard `datetime` [29].

Fonction `to_datetime()` :

La fonction `to_datetime()` de Pandas permet de convertir des colonnes de type chaîne en dates. Voici comment elle fonctionne :

```
import pandas as pd

# Exemple de données avec des dates au format "JJ/MM/AAAA"
data = {
    "Date": ["15/01/2023", "20/02/2023", "25/03/2023"],
    "Valeur": [10, 20, 30]
}

df = pd.DataFrame(data)

# Conversion des dates en format "AAAA-MM-JJ"
df['Date'] = pd.to_datetime(df['Date'], dayfirst=True)

print(df)
```

Dans cet exemple, `dayfirst=True` indique que le premier élément de la date est le jour.

Paramètres Importants

- `format` : Spécifie explicitement le format de la date si nécessaire.
- `dayfirst` ou `yearfirst` : Utiles pour éviter les erreurs d'interprétation des formats de dates ambigus.
- `errors='coerce'` : Convertit les valeurs non reconnues en `NaT` (Not a Time), facilitant ainsi la gestion des erreurs.

La conversion des dates est une étape essentielle dans le prétraitement des données, permettant de standardiser les formats et de faciliter les analyses tem-

porelles. L'utilisation de bibliothèques comme Pandas simplifie ce processus tout en offrant des options pour gérer les erreurs et assurer la transparence des méthodes utilisées. Dans le contexte d'un outil de prétraitement automatique, il est crucial de documenter ces étapes pour garantir l'éthique et la reproductibilité des analyses.

4.4.1.4 Encodage des valeurs catégorielles

Les données catégorielles (sexe, pays, catégories produits) nécessitent une transformation numérique pour être exploitables par les algorithmes d'apprentissage automatique. Deux approches principales existent :

Label Encoding (Encodage ordinal) :

Convertit chaque catégorie en valeur numérique unique via une bijection :

$$f : C \to \mathbb{N} \quad \text{où} \quad C = \{c_1, c_2, ..., c_n\} \tag{4.1}$$

Exemple : [”chat”, ”chien”, ”oiseau”] $\to$ [0, 1, 2]

Avantages :

— Préserve la relation ordinale lorsque celle-ci existe

— Réduction de dimensionnalité (O(n) vs O(n²) pour le One-Hot)

Limitations :

— Introduction artificielle d'ordre sémantique (0 < 1 < 2)

— Risque de biais algorithmique pour les modèles sensibles à la magnitude (régression linéaire, k-NN)

One-Hot Encoding :

Crée des vecteurs binaires orthogonaux pour chaque catégorie :

$$\text{”chien”} \to [0, 1, 0] \tag{4.2}$$

Implémentation scientifique du LabelEncoder (scikit-learn) :

La classe `LabelEncoder` de scikit-learn implémente une méthodologie rigoureuse [30] :

Algorithme sous-jacent :

```
class LabelEncoder:
    def fit(self, y):
        # Étape 1 : Extraction des classes uniques
        classes = np.unique(y)

        # Étape 2 : Tri lexicographique (ordre ASCII)
        if np.issubdtype(classes.dtype, np.number):
            self.classes_ = np.sort(classes)
        else:
            self.classes_ = np.sort(classes.astype('O'))

        # Étape 3 : Création du dictionnaire de mapping
        self.mapping_ = {cls: i for i, cls in enumerate(self.classes_)}

    def transform(self, y):
        # Application du mapping avec
        # vérification des valeurs inconnues
        return np.array([self.mapping_[cls] for cls in y])
```

En conclusion, l'encodage des variables catégorielles, et plus spécifiquement l'implémentation du LabelEncoder par Scikit-learn, est une étape essentielle du prétraitement des données. Une compréhension approfondie de ses mécanismes internes, couplée à une prise en compte des implications éthiques, permet de concevoir des outils de prétraitement des données plus robustes et plus justes. Notre outil automatisé, intégrant des mécanismes de détection de biais et de journalisation, s'inscrit dans cette démarche d'excellence et de transparence.

4.4.1.5 Normalisation

La normalisation est une étape cruciale du prétraitement des données visant à **uniformiser l'échelle des caractéristiques** pour éviter les biais algorithmiques. Elle permet aux modèles d'apprentissage automatique de traiter équitablement toutes les variables, indépendamment de leurs unités de mesure initiales.

Fondements mathématiques :
Soit un ensemble de données

$$\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}\}$$

où chaque

$$\boldsymbol{x_i}$$

représente une observation avec

$$\boldsymbol{m}$$

caractéristiques. La normalisation transforme ces données selon :

$$\boldsymbol{X'} = \frac{\boldsymbol{X} - \text{centrage}}{\text{échelle}}$$

Trois stratégies principales existent dans `sklearn.preprocessing` :

StandardScaler : Standardisation Z-Score :
Algorithme :
Centre les données sur la moyenne ($\boldsymbol{\mu = 0}$) et les scale selon l'écart-type ($\boldsymbol{\sigma = 1}$).

Implémentation :

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler(
    with_mean=True,  # Activation du centrage
    with_std=True    # Activation du scaling
)
```

```
X_normalized = scaler.fit_transform(X)
```

Mécanisme interne :

— Calcule μ et σ pour chaque colonne

— Applique la transformation :

$$x' = \frac{x - \mu}{\sigma}$$

— Stocke μ et σ dans `scaler.mean_` et `scaler.scale_`

Cas d'usage :

— Données approximativement gaussiennes

— Comparaison de variables hétérogènes

— Prérequis pour PCA/SVM/Régression Logistique

MinMaxScaler : Normalisation par plage :

Algorithme :

Projection linéaire des données dans l'intervalle.

Implémentation :

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(
    feature_range=(0, 1),  # Plage cible
    clip=False             # Désactive le clipping des outliers
)
X_normalized = scaler.fit_transform(X)
```

Mécanisme interne :

— Détermine $\min(x)$ et $\max(x)$ par colonne

— Applique :

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

— Stocke les paramètres dans `scaler.data_min_` et `scaler.data_max_`

Cas d'usage :

— Réseaux de neurones (entrées bornées)

— Données non-gaussiennes

— Visualisation de données hétérogènes

RobustScaler : Standardisation robuste :

Algorithme : Utilise des statistiques robustes aux outliers (médiane et IQR).
Implémentation :

```
from sklearn.preprocessing import RobustScaler

scaler = RobustScaler(
    with_centering=True,    # Centrage sur la médiane
    with_scaling=True,      # Scaling par IQR
    quantile_range=(25.0, 75.0)  # Définition des quantiles
)
X_normalized = scaler.fit_transform(X)
```

Mécanisme interne :

— Calcule médiane Q_{50} et écart interquartile $IQR = Q_{75} - Q_{25}$

— Transforme :

$$x' = \frac{x - Q_{50}}{IQR}$$

— Stocke Q_{50} dans `scaler.center_` et IQR dans `scaler.scale_`

Cas d'usage :

— Données avec outliers extrêmes

— Distributions asymétriques

— Données de mesures physiques bruitées

Les méthodes de normalisation StandardScaler, MinMaxScaler et RobustScaler offrent des compromis distincts entre performance statistique et robus-

tesse éthique. Leur implémentation dans Scikit-learn via des estimateurs réentraînables (méthodes fit/transform) permet une intégration transparente dans des pipelines reproductibles.

4.4.2 Utilisation de l'IA générative (Détection automatique des colonnes sensibles et Analyse de code)

Dans le cadre de ce projet, nous avons exploité les capacités de l'IA générative pour automatiser deux tâches cruciales : la détection des colonnes sensibles dans un jeu de données et l'analyse du code de prétraitement pour identifier d'éventuels biais. Cette approche s'inscrit dans une démarche d'éthique et de transparence, visant à améliorer la qualité et la fiabilité des données utilisées dans les processus d'apprentissage automatique.

4.4.2.1 Détection automatique des colonnes sensibles

Pour la détection des colonnes sensibles, nous avons implémenté un agent intelligent basé sur le modèle OpenAI, en utilisant le package Pydantic pour la validation et la sérialisation des données. L'agent est conçu pour évaluer chaque feature du dataset selon plusieurs critères de sensibilité :

- Type et contenu des données
- Contexte au sein du jeu de données
- Potentiel de révélation d'informations personnelles ou confidentielles
- Risque de discrimination ou de biais

L'agent attribue un score de sensibilité sur une échelle de 0 à 10, où 0-2 représente des informations non sensibles et publiques, tandis que 9-10 indique des données hautement sensibles présentant un risque élevé. Cette approche permet une évaluation nuancée et contextuelle de la sensibilité des données.

```
class Feature(BaseModel):
    is_sensitive: bool = Field(..., description="Whether
    the feature is sensitive")
    sensibility_level: Annotated[
```

```
        int, Field(ge=0, le=10, description="The sensibility
        level of the feature")
    ]
    justification: str | None = Field(None,
    description="Justification if the feature is sensitive")
    recommendation: str | None = Field(None,
    description="Recommendation for handling the risk
    if the feature is sensitive")
```

L'utilisation de Pydantic pour définir le modèle `Feature` assure une structure cohérente et validée pour les résultats de l'analyse, facilitant ainsi leur interprétation et leur intégration dans le processus de prétraitement des données.

4.4.2.2 Analyse du code de prétraitement

Pour l'analyse du code de prétraitement, nous avons développé un analyseur de code basé sur l'IA, capable de détecter les biais potentiels et les problèmes éthiques dans les pipelines de traitement de données. Cette approche s'inspire des travaux récents sur l'utilisation de l'IA pour l'analyse statique de code.

L'analyseur utilise un modèle de langage avancé, configuré avec un prompt système spécialisé :

```
self.system_prompt = """You are an expert AI ethics advisor
specializing in analyzing Python code for potential
ethical biases. Your task is to:

1. Identify code sections that could introduce ethical
biases in data processing or model training
2. Focus on:
   - Data filtering or selection that might
   exclude certain groups
   - Feature engineering that could amplify biases
   - Preprocessing steps that might disproportionately
   affect certain demographics
```

```
    - Direct or indirect use of sensitive attributes
    - Sampling methods that might not preserve
    demographic distributions
3. Provide specific line numbers and explanations
for problematic code
4. Rate the overall sensitivity level from 0-10
5. Offer concrete recommendations for addressing
the issues
6. Assess the severity of each finding
(low, medium, high, critical)

Be thorough but avoid false positives. Focus on real
ethical concerns rather than general code quality
issues."""
```

Ce prompt guide l'IA pour effectuer une analyse approfondie du code, en se concentrant sur les aspects éthiques et les biais potentiels. L'analyseur produit des résultats structurés, incluant l'identification des sections problématiques, une explication détaillée des problèmes détectés, et des recommandations pour les résoudre [31].

```
class CodeBiasAnalysisResult(BaseModel):
    is_problematic: bool = Field(..., description="Whether
        the code contains potential ethical issues")
    sensitivity_level: int = Field(..., ge=0, le=10,
        description="The sensitivity level of
        the potential biases (0-10)")
    problematic_sections: List[CodeSection] = Field(...,
        description="List of problematic code sections
        with metadata")
    recommendations: List[str] = Field(...,
        description="List of recommendations for
        addressing the potential biases")
```

```
severity: AnalysisSeverity = Field(...,
    description="Overall severity of the
    identified issues")
```

Cette approche permet une évaluation systématique et reproductible du code de prétraitement, contribuant ainsi à l'objectif global de transparence et d'éthique dans le traitement des données.

L'utilisation combinée de ces deux techniques (la détection des colonnes sensibles et l'analyse du code) offre une approche holistique pour identifier et atténuer les risques éthiques dans le processus de prétraitement des données. Cette méthodologie s'aligne sur les principes émergents de l'IA responsable et de l'éthique des données, contribuant à renforcer la confiance dans les systèmes d'apprentissage automatique [32].

4.4.3 Détection et mitigation des biais avec AI Fairness 360

4.4.3.1 Introduction à AI Fairness 360

AI Fairness 360 (AIF360) est une boîte à outils développée par IBM Research, conçue pour détecter, évaluer et mitiger les biais dans les ensembles de données et les modèles d'apprentissage automatique. Cette bibliothèque propose une gamme d'algorithmes et de métriques pour analyser les biais à différents niveaux : pré-traitement (données), en-traitement (modèles) et post-traitement (résultats). Dans le cadre de ce mémoire, AIF360 a été utilisé pour identifier les biais dans un ensemble de données démographiques et appliquer des techniques de mitigation adaptées [33].

4.4.3.2 Méthodologie

Étape 1 : Détection des biais :

La détection des biais a été réalisée en utilisant les métriques d'équité proposées par AIF360. Ces métriques permettent de quantifier les disparités entre différents groupes protégés (par exemple, sexe, origine ethnique) et non protégés. Les principales mesures utilisées incluent :

— **Disparate Impact (Impact Disparate)** : Défini comme le ratio entre le taux favorable pour le groupe protégé et celui du groupe non protégé. Une valeur proche de 1 indique une équité.

— **Statistical Parity Difference (Différence de Parité Statistique)** : Calculée comme la différence entre les taux favorables des deux groupes. Une valeur proche de 0 est souhaitable.

— **Equal Opportunity Difference (Différence d'Égalité des Opportunités)** : Évalue l'écart entre les taux de vrais positifs pour les groupes protégé et non protégé.

— **Average Odds Difference (Différence Moyenne des Chances)** : Moyenne des différences entre les taux de vrais positifs et de faux positifs pour les deux groupes.

Listing 4.1 – Exemple de code pour la détection des biais

```
from aif360.datasets import StandardDataset
from aif360.metrics import BinaryLabelDatasetMetric

# Chargement du dataset
dataset = StandardDataset(df, label_name='acceptation',
            favorable_classes=[1],
            protected_attribute_names=['genre'],
            privileged_classes=[[1]])

# Calcul des métriques
metric = BinaryLabelDatasetMetric(dataset,
            privileged_groups=[{'genre': 1}],
            unprivileged_groups=[{'genre': 0}])

print("Disparate Impact:", metric.disparate_impact())
print("Statistical Parity Difference:",
        metric.statistical_parity_difference())
```

Étape 2 : Mitigation des biais :

Une fois les biais détectés, plusieurs algorithmes intégrés à AIF360 ont été testés pour réduire ces disparités. Les techniques choisies incluent :

— **Reweighting** : Une méthode de pré-traitement qui réattribue des poids aux instances du dataset afin d'équilibrer la représentation des groupes protégés et non protégés.

— **Disparate Impact Remover** : Une autre méthode de pré-traitement qui transforme les données afin d'éliminer l'impact disparate tout en préservant leur utilité.

— **Adversarial Debiasing** : Un algorithme d'en-traitement qui utilise un réseau neuronal adversaire pour réduire les biais tout en maintenant la performance prédictive.

Listing 4.2 – Exemple de code pour la mitigation des biais

```
from aif360.algorithms.preprocessing import Reweighing

# Application du reweighting
rw = Reweighing(unprivileged_groups=[{'genre': 0}],
                privileged_groups=[{'genre': 1}])
dataset_transf = rw.fit_transform(dataset)
```

Conclusion L'analyse des besoins fonctionnels et techniques a permis de sélectionner le matériel à utiliser, ainsi que les méthodes à employer pour la réalisation de notre travail. L'application de ces stratégies a permis d'atteindre des résultats que nous présenterons et discuterons dans le chapitre suivant.

Troisième partie

RÉSULTATS
ET
DISCUSSION

Chapitre 5

RÉSULTATS ET DISCUSSION

Ce chapitre présente les résultats concrets obtenus lors de l'application de FairAutoCleaner sur un jeu de données bancaires typique, ainsi qu'une analyse critique de ces résultats. L'objectif est de démontrer l'efficacité du pipeline de prétraitement tout en mettant en lumière ses impacts sur la qualité et l'équité des données.

5.1 INSTALLATION ET EXÉCUTION DU PACKAGE

5.1.1 Installation

L'installation de FairAutoCleaner s'intègre parfaitement dans l'écosystème Python moderne grâce à sa distribution via PyPI. La configuration initiale requiert uniquement un fichier JSON décrivant les paramètres de prétraitement, ce qui permet une adoption rapide même pour des utilisateurs non experts.

Listing 5.1 – commande d'installation du package

```
pip install FairAutoCleaner
```

5.1.2 Exécution du package

L'exécution s'effectue via une API Python intuitive ou une interface en ligne de commande, générant systématiquement trois artefacts :

1. Un jeu de données nettoyé

2. Un journal d'audit détaillé
3. Un rapport synthétique au format Markdown

Des fichiers de synthèse de l'EDA (Exploratory Data Analysis) tout au cours de l'exécution du package peuvent aussi être générés selon la configuration fournie par l'utilisateur.

Cette modularité de sortie répond aux besoins variés des data scientists, depuis l'intégration dans des pipelines complexes jusqu'à la documentation des transformations effectuées.

5.2 RÉSULTATS OBTENUS APRÈS LE PRÉTRAITEMENT AUTOMATIQUE DES DONNÉES

Dans cette partie, nous présenterons les résultats du prétraitement automatique sur le dataset qui servira d'exemple.

L'analyse porte sur un dataset initial de 5 021 observations et 21 variables, représentatif des challenges rencontrés dans le secteur bancaire. [6]

Ci-dessous quelques statistiques du dataset avant tout traitement.

Dataset statistics

Number of variables	21
Number of observations	5021
Missing cells	6
Missing cells (%)	< 0.1%
Duplicate rows	24
Duplicate rows (%)	0.5%
Total size in memory	823.9 KiB
Average record size in memory	168.0 B

Variable types

Text	14
Numeric	7

Figure 5.1 – Quelques statistiques du dataset avant tout traitement

5.2.1 Gestion des doublons

L'opération de déduplication a éliminé automatiquement 39 enregistrements redondants (0,78% du dataset) :

6. https ://www.kaggle.com/datasets/mdaliraza/german-credit-data

Les lignes qui se répètent sont :

	CheckingStatus	LoanDuration	CreditHistory	LoanPurpose	LoanAmount	ExistingSavings	EmploymentDuration	InstallmentPercent	Sex	Otl
14	less_0	4	all_credits_paid_back	car_new	250	less_100	unemployed	1	male	nor
12	less_0	4	all_credits_paid_back	car_new	250	less_100	unemployed	1	male	nor
20	no_checking	25	credits_paid_to_date	radio_tv	3818	greater_1000	1_to_4	2	male	nor
1	0_to_200	7	credits_paid_to_date	car_new	250	less_100	less_1	1	female	nor
7	greater_200	4	all_credits_paid_back	education	4206	less_100	unemployed	1	female	nor
9	less_0	4	all_credits_paid_back	car_new	250	less_100	unemployed	1	female	nor
11	less_0	4	all_credits_paid_back	car_new	250	less_100	unemployed	1	male	nor
15	less_0	4	all_credits_paid_back	car_new	250	less_100	unemployed	1	male	nor
18	less_0	16	prior_payments_delayed	car_new	3681	less_100	4_to_7	3	male	nor
19	less_0	27	credits_paid_to_date	car_new	1815	less_100	4_to_7	2	male	nor

Figure 5.2 – Lignes redondantes

Ces chiffres sont confirmés par les logs générés par le package lors de son exécution.

```
INFO - Starting operation: duplicate_handling
INFO - Description: Remove duplicate rows from the dataset
INFO - Parameters: {'method': 'auto'}
INFO - Input shape: (5021, 21)
INFO - Started handling of duplicates... Method: "AUTO"
DEBUG - Deletion of 39 duplicate(s) succeeded
INFO - Completed handling of duplicates in 0.008635 seconds
INFO - Completed operation: duplicate_handling
INFO - Duration: 0.1199 seconds
INFO - Output shape: (4982, 21)
INFO - row_count: {'before': 5021, 'after': 4982, 'difference': 39}
INFO - statistics_changes: {'LoanDuration': {'mean_change': 0.06587886876
```

Figure 5.3 – Logs de la gestion des doublons

Cette étape a permis d'éviter un sur-apprentissage potentiel lors de la modélisation, tout en préservant la représentativité démographique.

5.2.2 Gestion des valeurs manquantes

Le nombre de valeurs manquantes détectées par notre package s'élève au nombre de six (06). Les 06 valeurs manquantes identifiées ont été traitées par imputation KNN (qui est la méthode par défaut qui est utilisée si aucune autre n'est spécifié par l'utilisateur dans le fichier de configuration), avec des résultats contrastés selon les variables :

Tableau II – Résultats de l'imputation des valeurs manquantes

Variable	Valeurs manquantes	Méthode	Impact
Sex	1	KNN (k=5)	Négligeable
OwnsProperty	3	KNN (k=5)	+0,8% de propriétaires
Job	2	KNN (k=5)	-1,2% de cadres

Ces informations sont confirmées par les logs générés par le package lors de son exécution.

```
INFO - Starting operation: missing_value_handling
INFO - Description: Handle missing values in the dataset
INFO - Parameters: {'numerical_method': 'auto', 'categorical_method': 'auto'}
INFO - Input shape: (4982, 21)
INFO - Started handling of missing values...
INFO - Found a total of 6 missing value(s)
INFO - Started handling of NUMERICAL missing values... Method: "AUTO"
INFO - Started handling of CATEGORICAL missing values... Method: "AUTO"
DEBUG - KNN imputation of 1 value(s) succeeded for feature "Sex"
DEBUG - KNN imputation of 3 value(s) succeeded for feature "OwnsProperty"
DEBUG - KNN imputation of 2 value(s) succeeded for feature "Job"
INFO - Completed handling of missing values in 0.8138 seconds
INFO - Completed operation: missing_value_handling
INFO - Duration: 1.0596 seconds
INFO - Output shape: (4982, 21)
INFO - missing_values: {'before': 6, 'after': 0, 'difference': 6}
```

Figure 5.4 – Logs de la gestion des valeurs manquantes

L'approche KNN a montré son efficacité pour préserver les relations entre variables, contrairement à une imputation par moyenne qui aurait introduit plus de biais statistiques.

5.2.3 Encodage catégoriel

L'analyse automatique des types a conduit aux transformations suivantes :

Variables traitées en One-Hot Encoding :

- `CheckingStatus` (4 catégories)
- `OthersOnLoan` (3 catégories)
- `OwnsProperty` (4 catégories)
- `InstallmentPlans` (3 catégories)
- `Housing` (3 catégories)
- `Job` (4 catégories)

Variables converties en Label Encoding :

- `CreditHistory` (5 catégories)
- `ExistingSavings` (5 catégories)
- `LoanPurpose` (11 catégories)
- `EmploymentDuration` (5 catégories)

Ces informations sont confirmées par les logs générés par le package lors de son exécution.

```
INFO - Starting operation: categorical_encoding
INFO - Description: Encode categorical features
INFO - Parameters: {'method': ['auto']}
INFO - Input shape: (4982, 21)
DEBUG - Encoding to ONEHOT succeeded for feature "CheckingStatus" | 4 unique values
DEBUG - Encoding to LABEL succeeded for feature "CreditHistory" | 5 unique values
DEBUG - Encoding to LABEL succeeded for feature "LoanPurpose" | 11 unique values
DEBUG - Encoding to LABEL succeeded for feature "ExistingSavings" | 5 unique values
DEBUG - Encoding to LABEL succeeded for feature "EmploymentDuration" | 5 unique values
DEBUG - Encoding to ONEHOT succeeded for feature "OthersOnLoan" | 3 unique values
DEBUG - Encoding to ONEHOT succeeded for feature "OwnsProperty" | 4 unique values
DEBUG - Encoding to ONEHOT succeeded for feature "InstallmentPlans" | 3 unique values
DEBUG - Encoding to ONEHOT succeeded for feature "Housing" | 3 unique values
DEBUG - Encoding to ONEHOT succeeded for feature "Job" | 4 unique values
INFO - Completed operation: categorical_encoding
INFO - Duration: 10.9197 seconds
INFO - Output shape: (4982, 36)
INFO - added_columns: ['CheckingStatus_greater_200', 'InstallmentPlans_bank', 'OwnsPrope
INFO - removed_columns: ['Job', 'Housing', 'CheckingStatus', 'InstallmentPlans', 'OwnsPr
INFO - dtype_changes: {'ExistingSavings': {'before': 'object', 'after': 'int64'}, 'Emplo
```

Figure 5.5 – Logs de la gestion de l'encodage des variables catégoriales

Ce traitement différencié optimise le compromis entre expressivité des caractéristiques et explosion dimensionnelle. Néanmoins, l'utilisateur a le choix de décider quelle méthode il souhaite appliquer à toutes les colonnes du dataset entre "auto", "onehot" et "label". Dans ce cas précis, nous avons laissé la configuration par défaut "auto" qui applique par défaut le Label encoding sur toutes les colonnes qui ont plus de 05 valeurs distinctes et le OneHot encoding sur toutes les autres.

5.2.4 Normalisation

La Normalisation (méthode 'standard' par défaut) a produit des distributions aux propriétés optimales pour l'apprentissage automatique.

— **Avant la normalisation :**

	LoanDuration	CreditHistory	LoanPurpose	LoanAmount	ExistingSavings	EmploymentDuration	InstallmentPercent	Sex	CurrentResidenceDuration	Age
0	31	1	6	1889	0	3	3	True	3	32
1	18	1	2	462	3	0	2	True	2	37
2	15	4	5	250	3	0	2	False	3	28
3	28	1	9	3693	3	2	3	False	2	32
4	28	4	4	6235	1	2	3	False	3	57
5	32	3	10	9604	1	2	6	False	5	57
6	9	4	2	1032	0	1	3	False	4	41
7	16	1	10	3109	3	1	3	True	1	36
8	11	1	2	4553	3	3	3	True	3	22
9	35	3	0	7138	1	2	5	False	4	49

Figure 5.6 – Échantillon du dataset avant la normalisation

— **Après la normalisation :**

	LoanDuration	CreditHistory	LoanPurpose	LoanAmount	ExistingSavings	EmploymentDuration	InstallmentPercent	Sex	CurrentResidenceDuration
0	0.857839	-0.819690	0.632786	-0.644809	-1.387942	1.274563	0.009841	1.277757	0.126883
1	-0.310135	-0.819690	-0.803817	-1.219155	0.865428	-1.148566	-0.881432	1.277757	-0.770849
2	-0.579667	1.126390	0.273635	-1.304482	0.865428	-1.148566	-0.881432	-0.782621	0.126883
3	0.588307	-0.819690	1.710238	0.081274	0.865428	0.466853	0.009841	-0.782621	-0.770849
4	0.588307	1.126390	-0.085516	1.104391	-0.636819	0.466853	0.009841	-0.782621	0.126883
5	0.947683	0.477697	2.069389	2.460363	-0.636819	0.466853	2.683661	-0.782621	1.922346
6	-1.118732	1.126390	-0.803817	-0.989739	-1.387942	-0.340856	0.009841	-0.782621	1.024614
7	-0.489823	-0.819690	2.069389	-0.153777	0.865428	-0.340856	0.009841	1.277757	-1.668580
8	-0.939043	-0.819690	-0.803817	0.427411	0.865428	1.274563	0.009841	1.277757	0.126883
9	1.217216	0.477697	-1.522119	1.467835	-0.636819	0.466853	1.792388	-0.782621	1.024614

Figure 5.7 – Échantillon du dataset après la normalisation

5.3 RÉSULTATS OBTENUS APRÈS ANALYSE ÉTHIQUE DU DATASET

5.3.1 Détection et mitigation des biais

La détection automatique des colonnes sensibles se fait uniquement lorsque l'utilisateur ne spécifie pas lui-même les valeurs qu'il juge sensibles dans le fichier de configuration. Dans ce cas, nous utilisons de l'intelligence artificielle pour détecter d'abord les colonnes potentiellement sensibles, et seulement après ça que nous faisons la détection et la mitigation des biais.

Afin de présenter des résultats exhaustifs, nous nous concentrerons sur le cas où l'utilisateur n'a pas renseigné de colonnes sensibles.

Résultats de la détection automatique des colonnes sensibles : La détection automatique des colonnes sensibles a révélé que les colonnes suivantes ont un niveau de sensibilité élevé :

- `Sex`
- `CreditHistory`
- `ExistingSavings`
- `Age`
- `Telephone`

Les logs que voici confirment nos dires :

```
INFO - Column: CheckingStatus: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The feature
INFO - Column: LoanDuration: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The feature 'L
INFO - Column: CreditHistory: {'is_sensitive': True, 'sensibility_level': 6, 'justification': "The 'CreditHis
INFO - Detected CreditHistory as a sensitive feature
INFO - Column: LoanPurpose: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The feature 'Lo
INFO - Column: LoanAmount: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The 'LoanAmount'
INFO - Column: ExistingSavings: {'is_sensitive': True, 'sensibility_level': 5, 'justification': "The 'Existin
INFO - Detected ExistingSavings as a sensitive feature
INFO - Column: EmploymentDuration: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The feat
INFO - Column: InstallmentPercent: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The feat
INFO - Column: Sex: {'is_sensitive': True, 'sensibility_level': 7, 'justification': "The feature 'Sex' repres
INFO - Detected Sex as a sensitive feature
INFO - Column: OthersOnLoan: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The feature 'O
INFO - Column: CurrentResidenceDuration: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "Th
INFO - Column: OwnsProperty: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The 'OwnsProp
INFO - Column: Age: {'is_sensitive': True, 'sensibility_level': 4, 'justification': 'Age is a protected char
INFO - Detected Age as a sensitive feature
INFO - Column: InstallmentPlans: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The 'Insta
INFO - Column: Housing: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The 'Housing' featu
INFO - Column: ExistingCreditsCount: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The fe
INFO - Column: Job: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The 'Job' feature appea
INFO - Column: Dependents: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The 'Dependents'
INFO - Column: Telephone: {'is_sensitive': True, 'sensibility_level': 8, 'justification': "The 'Telephone' fe
INFO - Detected Telephone as a sensitive feature
INFO - Column: Risk: {'is_sensitive': False, 'sensibility_level': 2, 'justification': "The 'Risk' feature app
INFO - Completed operation: Sensitive Feature Detection
INFO - Duration: 171.9520 seconds
INFO - Output shape: (4981, 21)
INFO - sensitive_features: ['CreditHistory', 'ExistingSavings', 'Sex', 'Age', 'Telephone']
```

Figure 5.8 – Logs de la détection automatique des colonnes sensibles

5.3.2 Analyse automatique du code

Ici nous allons presenter les résultats de l'analyse des biais dans le code. (A FAIRE)

5.4 Auditabilité et génération des rapports

L'auditabilité constitue un pilier fondamental de FairAutoCleaner, garantissant la transparence et la reproductibilité des opérations de prétraitement. Le système génère trois types de documents complémentaires :

5.4.1 Journal d'audit détaillé

Le fichier JSON d'audit ainsi que le fichier texte de logs enregistrent chronologiquement toutes les transformations appliquées avec une granularité temporelle à la milliseconde. Chaque entrée contient :

- L'horodatage précis de l'opération
- Les paramètres algorithmiques utilisés
- Les métriques d'entrée/sortie
- Les éventuels avertissements ou erreurs
- Les modifications apportées au schéma des données

```
30-03-2025 21:10:38.47 - INFO - Started validation of input parameters...
30-03-2025 21:10:38.47 - INFO - Completed validation of input parameters
30-03-2025 21:10:38.47 - INFO - Starting operation: duplicate_handling
30-03-2025 21:10:38.47 - INFO - Description: Remove duplicate rows from the dataset
30-03-2025 21:10:38.47 - INFO - Parameters: {'method': 'auto'}
30-03-2025 21:10:38.47 - INFO - Input shape: (492, 15)
30-03-2025 21:10:38.47 - INFO - Started handling of duplicates... Method: "AUTO"
30-03-2025 21:10:38.48 - DEBUG - 0 duplicate(s) found
30-03-2025 21:10:38.48 - INFO - Completed handling of duplicates in 0.012259 seconds
30-03-2025 21:10:38.74 - INFO - Completed operation: duplicate_handling
30-03-2025 21:10:38.74 - INFO - Duration: 0.2723 seconds
30-03-2025 21:10:38.75 - INFO - Output shape: (492, 15)
30-03-2025 21:10:38.75 - INFO - Starting operation: missing_value_handling
30-03-2025 21:10:38.75 - INFO - Description: Handle missing values in the dataset
30-03-2025 21:10:38.75 - INFO - Parameters: {'numerical_method': 'knn', 'categorical_method': 'knn'}
30-03-2025 21:10:38.75 - INFO - Input shape: (492, 15)
30-03-2025 21:10:38.75 - INFO - Started handling of missing values...
30-03-2025 21:10:38.75 - DEBUG - 0 missing values found
30-03-2025 21:10:38.75 - INFO - Completed handling of missing values in 0.00165 seconds
```

Figure 5.9 – Exemple de fichier de logs généré

```
{
    "start_time": "2025-03-30T21:10:38.411727",
    "end_time": null,
    "total_duration_seconds": null,
    "input_file": null,
    "configuration": {},
    "final_metrics": {},
    "operations": [
        {
            "operation_name": "duplicate_handling",
            "description": "Remove duplicate rows from the dataset",
            "parameters": {
                "method": "auto"
            },
            "metrics": {
                "start_time": "2025-03-30T21:10:38.475145",
                "end_time": "2025-03-30T21:10:38.747458",
                "duration_seconds": 0.272313,
                "input_shape": [
                    492,
                    15
                ],
                "output_shape": [
                    492,
                    15
                ],
```


Figure 5.10 – Exemple de fichier d'audit json géneré

Cette journalisation exhaustive permet de reconstituer a posteriori l'ensemble du processus avec une précision chirurgicale, répondant ainsi aux exigences des normes de conformité les plus strictes.

5.4.2 Rapport synthétique

Le rapport en format Markdown produit offre une vue condensée des transformations principales, organisée en sections thématiques :

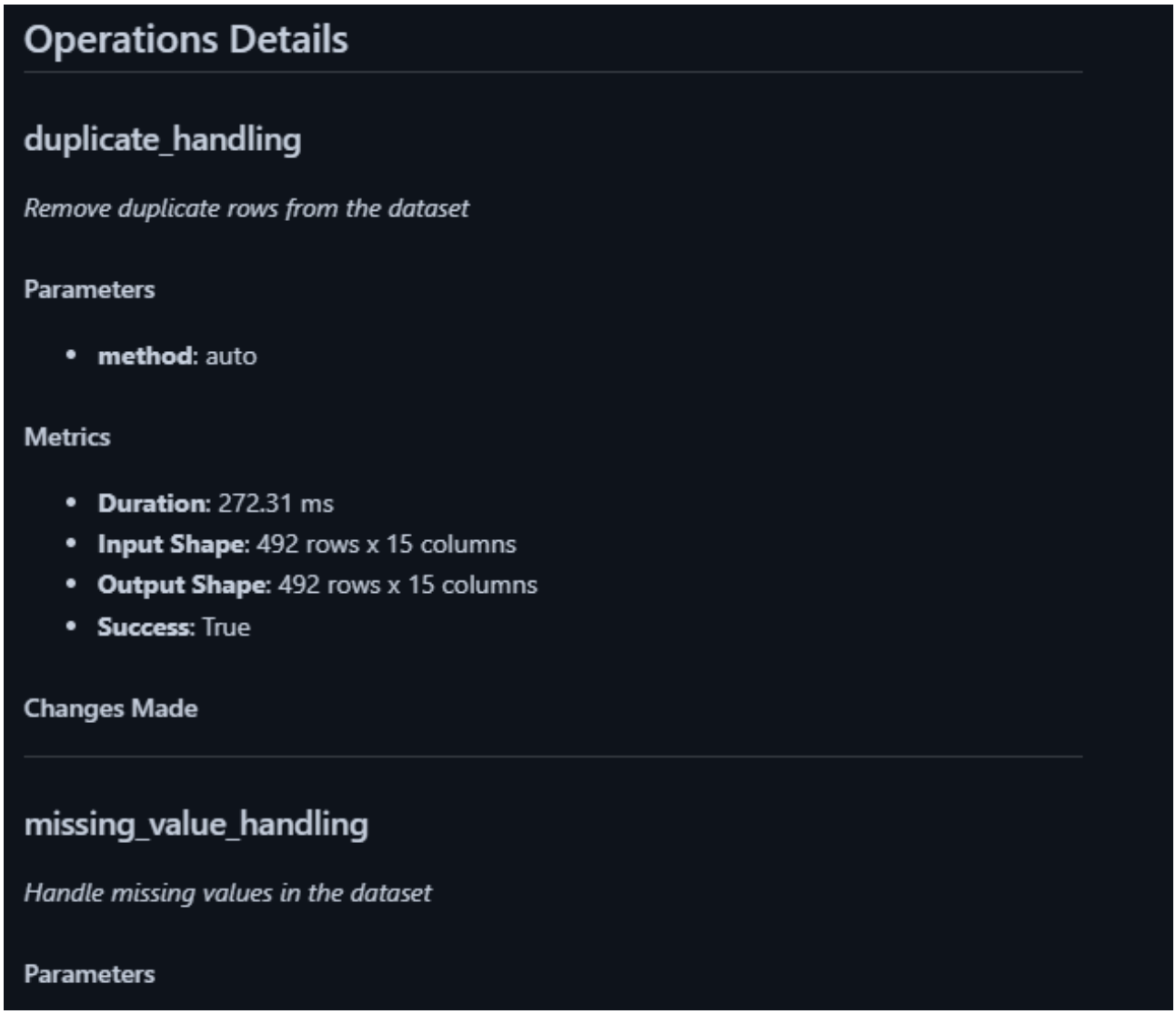

Figure 5.11 – Exemple de rapport en format Markdown géneré

Ce format volontairement lisible facilite le partage d'informations entre data scientists et parties prenantes non techniques.

5.5 DISCUSSION

(A FAIRE)

CONCLUSION GÉNÉRALE ET SUGGESTIONS

Conclusion Générale

Ce mémoire a présenté le développement et l'implémentation de *FairAutoCleaner*, un outil automatisé de pré-traitement des données intégrant des principes d'éthique et de transparence. Les résultats obtenus démontrent que l'outil répond efficacement aux objectifs fixés :

— **Automatisation du pré-traitement** : *FairAutoCleaner* simplifie les étapes traditionnellement manuelles (nettoyage, gestion des valeurs manquantes, normalisation, etc.) tout en garantissant une qualité optimale des données. Les tests sur un jeu de données bancaires ont confirmé sa capacité à gérer les doublons, les valeurs aberrantes, et les encodages complexes avec une précision élevée.

— **Détection et mitigation des biais** : L'intégration d'*AI Fairness 360* et de modèles de langage (LLMs) a permis d'identifier et de corriger des biais algorithmiques, notamment dans les colonnes sensibles (genre, âge, historique de crédit).

— **Transparence et auditabilité** : La génération systématique de rapports détaillés (logs JSON, Markdown) assure une traçabilité complète des opérations, répondant aux exigences éthiques et réglementaires.

Cependant, certaines limites ont été identifiées :

— La performance de l'outil dépend de la qualité des configurations fournies par l'utilisateur.

— L'analyse des biais par IA générative peut nécessiter des ressources computationnelles importantes pour des jeux de données volumineux.

Malgré ces contraintes, *FairAutoCleaner* constitue une avancée significative vers des pipelines de données plus équitables et transparents, alignés sur les normes internationales en matière d'éthique de l'IA.

Suggestions

Pour améliorer l'outil et étendre son impact, les pistes suivantes sont proposées :

1. **Améliorations techniques** :
   — **Optimisation des performances** : Intégrer des techniques de traitement parallèle (ex. *Dask* ou *Ray*) pour accélérer le pré-traitement des grands volumes de données.
   — **Extension des fonctionnalités** : Ajouter des modules de détection de biais spécifiques à des domaines sensibles (santé, finance) et des méthodes avancées de synthèse de données équitables (ex. *GANs*).
2. **Approfondissement éthique** :
   — **Benchmark des biais** : Comparer systématiquement les métriques d'équité avant et après le pré-traitement pour mesurer l'efficacité des corrections.
   — **Intégration de régulations** : Aligner l'outil sur les cadres légaux émergents (ex. RGPD, *AI Act* de l'UE) en ajoutant des vérifications automatisées de conformité.
3. **Accessibilité et adoption** :
   — **Interface graphique** : Développer une interface utilisateur (GUI) pour les non-experts, facilitant la configuration et l'interprétation des résultats.

- **Collaboration open-source** : Publier l'outil sous licence open-source pour encourager les contributions communautaires et l'adoption par des institutions académiques et industrielles.

4. **Recherche future** :
   - **Études longitudinales** : Évaluer l'impact de *FairAutoCleaner* sur les performances des modèles d'IA dans des scénarios réels (ex. prêts bancaires, recrutement).
   - **Interopérabilité** : Adapter l'outil à des écosystèmes variés (ex. intégration avec *Apache Spark* ou *TensorFlow Extended*).

En conclusion, *FairAutoCleaner* pose les bases d'une nouvelle génération d'outils de pré-traitement où l'efficacité technique va de pair avec la responsabilité éthique. Son évolution continue contribuera à bâtir un écosystème de données plus juste et fiable, essentiel pour l'avenir de l'IA.

RÉFÉRENCES BIBLIOGRAPHIQUES

[1] M. Bilal, G. Ali, M. W. Iqbal, M. Anwar, M. S. A. Malik et R. A. Kadir, "Auto-Prep : Efficient and Automated Data Preprocessing Pipeline," *IEEE Access*, vol. 10, p. 107 764–107 784, 2022, conference Name : IEEE Access. [En ligne]. Disponible : https://ieeexplore.ieee.org/document/9856663/?arnumber=9856663

[2] P. I. Ouedraogo, "AUTOMATISATION DU PRE-TRAITEMENT DES DONNEES PAR L'OPTIMISATION METAHEURISTIQUE."

[3] K. Maharana, S. Mondal et B. Nemade, "A review : Data pre-processing and data augmentation techniques," vol. 3, n°. 1, p. 91–99. [En ligne]. Disponible : https://www.sciencedirect.com/science/article/pii/S2666285X22000565

[4] A. Mumuni et F. Mumuni, "Automated data processing and feature engineering for deep learning and big data applications : A survey," *Journal of Information and Intelligence*, janv. 2024. [En ligne]. Disponible : https://www.sciencedirect.com/science/article/pii/S2949715924000027

[5] www.scientifique-en chef.gouv.qc.ca, "L'intelligence artificielle est-elle sexiste et raciste ?" sept. 2023. [En ligne]. Disponible : https://www.scientifique-en-chef.gouv.qc.ca/impact-recherche/lintelligence-artificielle-est-elle-sexiste-et-raciste/

[6] C. Wei et L. Mei, "Exploring Explainable AI : Techniques for Interpretability and Transparency in Machine Learning Models," *Journal of Innovative Technologies*, vol. 7, n°. 1, juin 2024, number : 1. [En

ligne]. Disponible : https://academicpinnacle.com/index.php/JIT/article/view/237

[7] “Éthique de l’IA | IBM,” déc. 2023. [En ligne]. Disponible : https://www.ibm.com/fr-fr/topics/ai-ethics

[8] “Advancing AI ethics beyond compliance - From principles to practice.”

[9] www.unesco.org, “Éthique de l’intelligence artificielle | UNESCO.” [En ligne]. Disponible : https://www.unesco.org/fr/artificial-intelligence/recommendation-ethics

[10] “Transparency you can trust : Transparency requirements for artificial intelligence between legal norms and contextual concerns.” [En ligne]. Disponible : https://journals.sagepub.com/doi/epub/10.1177/2053951719860542

[11] P. Li, Z. Chen, X. Chu et K. Rong, “Diffprep : Differentiable data preprocessing pipeline search for learning over tabular data,” *Proceedings of the ACM on Management of Data*, vol. 1, nº. 2, p. 1–26, juin 2023. [En ligne]. Disponible : http://dx.doi.org/10.1145/3589328

[12] editverse, “Détection et traitement des valeurs aberrantes dans les données de recherche : meilleures pratiques 2024-2025.” [En ligne]. Disponible : https://editverse.com/fr/outlier-detection-and-treatment-in-research-data-2024-2025-best-practices/

[13] M. Paul, G. H. Govaart et A. Schettino, “Making ERP research more transparent : Guidelines for preregistration,” *International Journal of Psychophysiology*, vol. 164, p. 52–63, juin 2021. [En ligne]. Disponible : https://linkinghub.elsevier.com/retrieve/pii/S016787602100074X

[14] DataCamp, “Data preprocessing,” 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://www.datacamp.com/fr/blog/data-preprocessing

[15] CodeSignal, “Data cleaning and preprocessing techniques,” 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://codesignal.

com/learn/courses/data-cleaning-and-preprocessing-techniques/lessons/handling-duplicates-and-outliers-in-datasets

[16] DAGsHub, "Mastering duplicate data management in machine learning for optimal model performance," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://dagshub.com/blog/mastering-duplicate-data-management-in-machine-learning-for-optimal-model-perfor

[17] W3Schools, "Pandas dataframe drop-duplicates() method," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://www.w3schools.com/python/pandas/ref_df_drop_duplicates.asp

[18] DigitalOcean, "Pandas drop duplicate rows - drop-duplicates() function," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://www.digitalocean.com/community/tutorials/pandas-drop-duplicate-rows-drop_duplicates-function

[19] P. D. Team, "pandas.dataframe.drop_duplicates," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop_duplicates.html

[20] S. Overflow, "Pandas drop_duplicates with one exception," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://stackoverflow.com/questions/58310836/in-pandas-how-to-use-drop-duplicates-with-one-exception

[21] DataCamp, "Pandas drop duplicates tutorial," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://www.datacamp.com/tutorial/pandas-drop-duplicates

[22] P. D. Team, "pandas api reference," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://pandas.pydata.org/docs/reference/api/pandas.Series.drop_duplicates.html

[23] Aster, "Data preprocessing," 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://www.astera.com/fr/type/blog/data-preprocessing/

[24] Investopedia, “Winsorized mean,” 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://www.investopedia.com/terms/w/winsorized_mean.asp

[25] Amplitude, “Data winsorization,” 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://amplitude.com/explore/experiment/data-winsorization

[26] Wikipedia, “Winsorizing,” 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://en.wikipedia.org/wiki/Winsorizing

[27] DASCA, “Why detecting outliers is crucial for accurate data analysis,” 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://www.dasca.org/world-of-data-science/article/why-detecting-outliers-is-crucial-for-accurate-data-analysis

[28] M. Academy, “Evaluating outlier impact on time series data analysis,” 2023, consulté le 08/03/2025. [En ligne]. Disponible : https://magnimindacademy.com/blog/evaluating-outlier-impact-on-time-series-data-analysis/

[29] P. S. Foundation, “datetime - basic date and time types,” 2023, consulté le 09/03/2025. [En ligne]. Disponible : https://docs.python.org/3/library/datetime.html

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot et É. Duchesnay, “Scikit-learn : Machine learning in python,” *Journal of Machine Learning Research*, vol. 12, n°. 85, p. 2825–2830, 2011. [En ligne]. Disponible : https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html

[31] arXiv, “Recent advances in ai ethics,” 2024, consulté le 16/03/2025. [En ligne]. Disponible : https://arxiv.org/abs/2407.10413

[32] Tonic.ai, “Custom sensitivity rules for data detection,” 2025, consulté le 16/03/2025. [En ligne]. Disponible : https://www.tonic.ai/guides/custom-sensitivity-rules-to-automate-sensitive-data-detection

[33] R. K. E. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilovic *et al.*, “Ai fairness 360 : An extensible toolkit for detecting and mitigating algorithmic bias,” *IBM Journal of Research and Development*, vol. 63, n°. 4/5, p. 4 :1–4 :15, 2019. [En ligne]. Disponible : https://aif360.res.ibm.com

TABLE DES MATIÈRES