REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
الجمهورية الجزائرية الديمقراطية الشعبية

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

HIGHER SCHOOL IN APPLIED SCIENCES
– T L E M C E N –

المدرسة العليا في العلوم التطبيقية
École Supérieure en Sciences Appliquées

وزارة التعليم العالي والبحث العلمي
المدرسة العليا في العلوم التطبيقية
- تلمسان -

Mémoire de fin d'étude

Pour l'obtention du diplôme de Master

Filière : Électronique
Option : Informatique Industrielle

Présenté par : Aya RIGHI

Thème

Classification intelligente des états d'un système de détection des fuites d'eau par apprentissage automatique

Soutenu publiquement, le 29 / 09 / 2026, devant le jury composé de :

Mme Nassima BREKSI REGUIG	MCB	ESSA. Tlemcen	Présidente
M Anisse CHIALI	MCB	ESSA. Tlemcen	Directeur de mémoire
Mme Nadjia MIDOUN	MCB	ESSA. Tlemcen	Examinatrice 1
Mme Asma BELAID	MCB	ESSA. Tlemcen	Examinatrice 2
Mme Yasmine KADAOUI	MCB	ESSA. Tlemcen	partenaire socio-économique
M. Abdou BOUAYAD			Ingénieur ADE

Année universitaire : 2025 /2026

Dédicace

À ma mère,
celle qui a toujours été présente à mes côtés, depuis mes premiers pas jusqu'à l'aboutissement de ce parcours. Pour son amour immense, ses sacrifices, sa patience, ses prières, ses encouragements et toute la force qu'elle m'a transmise au fil des années. Elle a partagé mes joies, mes difficultés, mes doutes et mes réussites, sans jamais cesser de croire en moi. Une grande partie de ce que je suis aujourd'hui, de ma persévérance et de ma détermination, je la lui dois.

À mon père,
pour sa présence constante tout au long de ma vie, pour ses sacrifices, ses conseils, sa confiance et le soutien qu'il m'a toujours apporté. Il a été un repère dans mon parcours, m'encourageant à avancer, à croire en mes capacités et à poursuivre mes objectifs avec sérieux et détermination. Ce travail représente également le fruit de tout ce qu'il m'a transmis au fil des années.

À ma petite famille,
pour votre amour, votre présence, vos encouragements et votre soutien de chaque instant. Vous avez toujours été une source de force, de réconfort et de motivation tout au long de mon parcours.

À Saim Ahmed Ramzi,
malgré une présence relativement récente dans ma vie, il est rapidement devenu l'un de mes soutiens les plus précieux après ma famille. Sa présence, ses encouragements, sa bienveillance et son soutien constant ont occupé une place importante dans cette étape de ma vie.

À mes collègues de la première promotion en Électronique,
avec qui j'ai partagé une étape importante de notre parcours, marquée par le travail, les défis et des souvenirs qui resteront gravés.

Je dédie ce travail à toutes les personnes qui ont accompagné mon parcours et qui ont contribué, chacune à leur manière, à faire de cette réussite une réalité.

Remerciements

Avant toute chose, je remercie **Allah, le Tout-Puissant**, de m'avoir accordé la force, la patience et la persévérance nécessaires pour mener ce travail à son terme.

Je tiens également à me remercier pour les efforts accomplis, la patience, la persévérance et la détermination dont j'ai fait preuve tout au long de ce parcours. Malgré les difficultés, les périodes de fatigue et les nombreuses remises en question, j'ai continué à avancer avec sérieux et engagement. Je me remercie d'avoir su apprendre, m'adapter, corriger et persévérer jusqu'à l'aboutissement de ce travail. Ce mémoire représente ainsi non seulement le résultat d'un travail académique, mais aussi le reflet d'un parcours personnel marqué par l'effort, la progression et la volonté d'aller jusqu'au bout.

Je tiens à exprimer ma profonde gratitude à **Mme Amina Ouhoud** pour son accompagnement, sa disponibilité, son implication et le soutien qu'elle m'a apporté tout au long de la réalisation de ce mémoire. Ses orientations, ses remarques et son suivi ont largement contribué à l'avancement et à l'aboutissement de ce travail.

J'adresse mes sincères remerciements à mon encadrant, **M. Anisse CHIALI**, pour son accompagnement, ses conseils, ses orientations ainsi que pour la confiance qu'il m'a accordée. Son encadrement et ses remarques ont contribué à faire progresser ce travail et à orienter sa réalisation dans de bonnes conditions.

Je souhaite aussi exprimer une reconnaissance particulière à **M. Saim Ahmed Ramzi, ingénieur en électronique**, pour sa présence constante, son soutien moral, sa disponibilité et son accompagnement tout au long de cette période. Son aide dans la mise en page et la finalisation de ce mémoire, ainsi que sa présence aux différentes étapes de ce travail, m'ont été particulièrement précieuses.

Je tiens à exprimer mes sincères remerciements à **Mme Nassima BREKSI REGUIG**, Présidente du jury, ainsi qu'à **Mme Nadjia MIDOUN** et **Mme Asma BELAID**, Examinatrices, à **Mme Yasmine KADAOUI**, Directrice de l'incubateur, et à **M. Abdou BOUAYAD**, Ingénieur ADE, pour l'honneur qu'elles et qu'ils me font en acceptant d'évaluer ce travail et pour l'attention qu'ils lui accorderont.

Enfin, j'adresse mes remerciements à toutes les personnes qui ont contribué, directement ou indirectement, à la réalisation de ce mémoire.

ملخص

يركز هذا العمل على تطبيق تقنيات التعلم الآلي في كشف وتصنيف تسربات المياه اعتماداً على بيانات هيدروليكية تجريبية. تم إنشاء قاعدة بيانات انطلاقاً من نموذج هيدروليكي مجهز بأجهزة قياس، بالاعتماد على التدفق والضغط ومستوى الخزان وحالة المضخة. بعد تحضير البيانات وتحليلها، تم اعتماد نموذج الغابة العشوائية Random Forest مع Optuna تحسينه باستخدام Optuna. ثم دمجه في تطبيق مراقبة يعتمد على Flask. أظهرت النتائج إمكانية تطبيق هذا النهج على النموذج التجريبي.

الكلمات المفتاحية : كشف التسربات، التعلم الآلي، الغابة العشوائية، البيانات الهيدروليكية، المراقبة.

Résumé

Ce travail porte sur l'application de l'apprentissage automatique à la détection et à la classification des fuites d'eau à partir de données hydrauliques expérimentales. Une base de données a été constituée à partir d'une maquette instrumentée en exploitant le débit, la pression, le niveau du réservoir et l'état de la pompe. Après préparation et analyse des données, un modèle de classification de type Random Forest, optimisé avec Optuna, a été retenu puis intégré à une application Flask de supervision. Les résultats obtenus montrent la faisabilité de cette approche sur la maquette expérimentale.

Mots-clés : détection de fuites, apprentissage automatique, Random Forest, données hydrauliques, supervision.

Abstract

This work focuses on the application of machine learning to water leak detection and classification using experimental hydraulic data. A dataset was built from an instrumented hydraulic prototype using flow rate, pressure, reservoir level, and pump status. After data preparation and analysis, a Random Forest classifier optimized with Optuna was selected and integrated into a Flask-based monitoring application. The obtained results demonstrate the feasibility of this approach on the experimental prototype.

Keywords: water leak detection, machine learning, Random Forest, hydraulic data, monitoring.

Liste des abréviations

ANN : Artificial Neural Network (réseau de neurones artificiels)

CNN : Convolutional Neural Network (réseau de neurones convolutif)

DL : Deep Learning (apprentissage profond)

DMA : District Metered Area (secteur de réseau instrumenté)

DT : Decision Tree (arbre de décision)

EPANET : Environmental Protection Agency Network Evaluation Tool

ESP32 : famille de microcontrôleurs d'Espressif Systems

FN : Faux négatif

FP : Faux positif

GAN : Generative Adversarial Network (réseau antagoniste génératif)

GNN : Graph Neural Network (réseau de neurones sur graphes)

IA : Intelligence artificielle

IoT : Internet of Things (Internet des objets)

k-NN : k-Nearest Neighbors (k plus proches voisins)

LightGBM : Light Gradient Boosting Machine

LSTM : Long Short-Term Memory

ML : Machine Learning (apprentissage automatique)

RBF : Radial Basis Function (fonction à base radiale)

RF: Random Forest (forêt aléatoire)

SCADA: Supervisory Control and Data Acquisition

SVM : Support Vector Machine (machine à vecteurs de support)

VN : Vrai négatif

VP : Vrai positif

WDS : Water Distribution System (réseau de distribution d'eau)

XGBoost : eXtreme Gradient Boosting

Liste des figures

Figure 1.1– Positionnement hiérarchique de l'intelligence artificielle, de l'apprentissage automatique et de l'apprentissage profond. 2
Figure 1.2– Chaîne générale d'un système d'apprentissage automatique, de l'acquisition des données à la prédiction. 3
Figure 1.3– Principales catégories d'apprentissage automatique. 4
Figure 1.4– Séparation des données en ensembles d'apprentissage, de validation et de test. ... 8
Figure 1.5– Principe d'un Arbre de décision. 9
Figure 1.6– Principe du k-NN : classification par vote majoritaire des k plus proches voisins. 10
Figure 1.7– Fonction logistique utilisée pour convertir une combinaison linéaire de variables en probabilité 10
Figure 1.8– Principe de la forêt aléatoire : agrégation des décisions de plusieurs arbres entraînés sur des échantillons bootstrap. 11
Figure 1.9– Principe du SVM : hyperplan à marge maximale et vecteurs de support. 12
Figure 1.10– Structure générale d'une matrice de confusion binaire. 14
Figure 1.11– Illustration conceptuelle du compromis biais-variance en fonction de la complexité du modèle 16
Figure 3.1— Structure initiale de la base de données dans le notebook 27
Figure 3.2– Structure de la base après suppression des doublons. 28
Figure 3.3– Vérification des valeurs manquantes dans la base. 29
Figure 3.4– Répartition des huit classes après suppression des doublons 30
Figure 3.5– Matrice de corrélation de Pearson des variables d'entrée. 31
Figure 3.6–Répartition des classes après équilibrage par SMOTE. 33
Figure 3.7–Capture – Résultats de la Random Forest de référence sur le jeu de test. 34
Figure 3.8–Capture – Classement des meilleurs essais obtenus avec Optuna. 36
Figure 3.9–Capture – Configuration du meilleur modèle sélectionné par Optuna. 36
Figure 3.10— Résultat du modèle final retenu après optimisation 37
Figure 3.11– Matrice de confusion de la Random Forest issue d'Optuna. 38
Figure 3.12– content/smartleak_ml_package.joblib 39
Figure 3.13– Vérification des performances après rechargement du modèle. 39

Figure 3.14– Validation de la communication entre le client de test, le serveur Flask et le modèle de classification. 41
Figure 3.15— Validation du scénario de fonctionnement normal sur le dashboard. 41
Figure 3.16— Validation du scénario de fuite importante sur l'interface de supervision 42
Figure 3.17– Validation du scénario de fuite faible sur l'interface de supervision 42
Figure 3.18–Validation des scénarios NORMAL + RESERVOIR_VIDE et FUITE_FAIBLE + RESERVOIR_VIDE. 43
Figure 3.19– Affichage du scénario FUITE_FAIBLE + RESERVOIR_VIDE sur le dashboard. 43
Figure 3.20— Exemple de non-concordance entre l'état attendu et la classe prédite. 44
Figure 3.21– – Diagramme de classe UML représentant la structure d'une observation expérimentale. 54
Figure 3.22– google Colab 54
Figure 3.23– Vs Code..... 55

Liste des tableaux

Tableau 1-1– Comparaison des principales catégories d'apprentissage automatique. 4
Tableau 1-2– Synthèse comparative de cinq algorithmes classiques de classification. 13
Tableau 1-3 Synthèse des principales métriques de classification. 15
Tableau 2-1 Synthèse des principales méthodes conventionnelles de détection des fuites 20
Tableau 2-2 Synthèse comparative des travaux récents sur la détection des fuites d'eau par intelligence artificielle. 21
Tableau 3-1–Organisation générale de la démarche expérimentale et de modélisation 26
Tableau 3-2– Variables utilisées pour la classification. 27
Tableau 3-3–Effectifs des classes dans la base nettoyée. 29
Tableau 3-4 – Statistiques descriptives des variables numériques 30
Tableau 3-5–Répartition des classes avant et après SMOTE. 32
Tableau 3-6– Dimensions des ensembles utilisés pour l'apprentissage et le test. 33
Tableau 3-7– Hyperparamètres de la Random Forest de référence. 33
Tableau 3-8– Hyperparamètres du meilleur essai Optuna. 35
Tableau 3-9 – Comparaison des performances sur le jeu de test. 36
Tableau 3-10— Rapport de classification du modèle Random Forest issu d'Optuna. 38
Tableau 3-11— Scénarios concordants utilisés pour la validation fonctionnelle. 40

Tableau des matières

Dédicace I
Remerciements II
ملخص III
Résumé III
Abstract III
Liste des abréviations IV
Liste des figures V
Liste des tableaux VII
Tableau des matières VIII
Introduction Générale I
Chapitre 1 Fondements théoriques de l'apprentissage automatique 1
Introduction 2
1.1 Introduction à l'intelligence artificielle 2
1.1.1 Notions essentielles sur l'apprentissage profond 3
1.2 Principes généraux et principales approches d'apprentissage automatique 3
1.3 Apprentissage supervisé : principes et tâches 5
1.3.1 Principe de l'apprentissage supervisé 5
1.3.2 Algorithmes de régression 6
1.4 Prétraitement des données 6
1.4.1 Nettoyage des données 6
1.4.2 Normalisation et standardisation 6
1.4.3 Encodage des variables catégorielles 7
1.4.4 Partitionnement des données et validation du modèle 7
1.5 Principaux algorithmes de classification 8
1.5.1 Arbre de décision 8
1.5.2 K plus proches voisins (k-NN) 9
1.5.3 Régression logistique 10
1.5.4 Forêt aléatoire (Random Forest) 11
1.5.5 Machine à vecteurs de support (SVM) 12
1.6 Critères d'évaluation 13
1.6.1 Matrice de confusion 13
1.6.2 Accuracy, Precision et Recall 14
1.6.3 F1-score 15
1.7 Surapprentissage et sous-apprentissage 15
1.7.1 Définitions et causes 15

1.7.2 Compromis biais-variance 16
1.7.3 Moyens de détection et de mitigation 16
Conclusion 16
Chapitre 2 Intelligence artificielle appliquée à la détection des fuites d'eau : état de l'art 18
Introduction 19
2.1 Des méthodes conventionnelles aux approches basées sur les données 19
2.1.1 Méthodes conventionnelles de détection 19
2.2 Rôle de l'intelligence artificielle et données exploitées 20
2.2.1 Tâches du diagnostic intelligent 20
2.2.2 Données hydrauliques, acoustiques et origine des observations 20
2.3 État de l'art et synthèse comparative des approches récentes 21
2.4 Positionnement du présent travail 24
Conclusion 24
Chapitre 3 Préparation des données, apprentissage et validation du système de classification 25
Introduction 26
3.1 Démarche générale du travail 26
3.2 Base de données expérimentale 27
3.2.1 Structure des données 27
3.2.2 Contrôle des données 27
3.2.3 Répartition des états de fonctionnement 29
3.3 Analyse exploratoire des données 30
3.3.1 Statistiques descriptives 30
3.3.2 Analyse des corrélations entre les variables d'entrée 31
3.4 Préparation des données pour l'apprentissage 31
3.4.1 Encodage des variables catégorielles 32
3.4.2 Équilibrage des classes par SMOTE 32
3.4.3 Standardisation et séparation apprentissage-test 33
3.5 Modèle de référence : Random Forest 33
3.6 Optimisation des modèles avec Optuna 35
3.6.1 Protocole d'optimisation 35
3.6.2 Configuration retenue par Optuna 35
3.7 Évaluation et interprétation des résultats 36
3.7.1 Comparaison globale des deux configurations 36
3.7.2 Choix final 37
3.7.3 Analyse par classe et matrice de confusion 37

3.8 Sauvegarde et intégration du modèle 39
3.9 Validation fonctionnelle du système développé 39
3.9.1 Principe de validation 39
3.9.2 Vérification de la communication et de l'affichage 40
3.9.3 Vérification des états associés au réservoir vide 42
3.9.4 Portée de la validation fonctionnelle 43
3.10 Discussion critique et limites 44
3.10.1 Limite liée à l'ordre du prétraitement 44
3.10.2 Exemple de non-concordance 44
3.10.3 Représentativité et généralisation 45
3.10.4 Apport du travail réalisé 45
Perspectives 45
Conclusion 46
Conclusion générale 47
Références bibliographiques 48
Annexes 53
Google Colab 54
Visual Studio Code 55

Introduction Générale

L'eau constitue une ressource essentielle dont la gestion représente un enjeu majeur sur les plans technique, économique et environnemental. Dans les systèmes hydrauliques, les fuites peuvent engendrer des pertes importantes, diminuer les performances des installations et accroître les coûts d'exploitation et de maintenance.

La détection de ces anomalies reste cependant complexe, car les variations observées au niveau du débit et de la pression ne sont pas nécessairement dues à une fuite et peuvent également résulter de conditions normales de fonctionnement. Cette difficulté justifie le recours à des approches capables d'exploiter simultanément plusieurs grandeurs afin d'améliorer l'identification des différents états du système.

Dans ce contexte, l'intelligence artificielle, et plus particulièrement l'apprentissage automatique, constitue une approche pertinente pour analyser les données acquises et reconnaître automatiquement des comportements caractéristiques. Le présent travail repose ainsi sur l'exploitation de données expérimentales acquises à partir d'un prototype instrumenté dédié à la détection des fuites d'eau.

L'objectif principal est de développer une approche de classification permettant de distinguer différents états de fonctionnement à partir de variables telles que le débit, la pression, le niveau d'eau et l'état de la pompe. Le travail associe ainsi l'expérimentation, le traitement des données et l'apprentissage automatique dans une démarche commune de surveillance du système.

Le mémoire est organisé en trois chapitres :

Chapitre 1 présente les fondements théoriques nécessaires à la compréhension de l'apprentissage automatique.

Chapitre 2 expose les principales approches appliquées à la détection des fuites d'eau et situe le travail par rapport aux recherches existantes.

Chapitre 3 présente la démarche expérimentale, la préparation des données, la mise en œuvre de la classification, l'évaluation du modèle ainsi que sa validation dans le système développé.

Chapitre 1

Fondements théoriques de l'apprentissage automatique

Introduction

Ce chapitre présente les fondements théoriques nécessaires à la compréhension de l'apprentissage automatique appliqué aux problèmes de classification. Il situe d'abord l'apprentissage automatique au sein de l'intelligence artificielle, puis expose les principales formes d'apprentissage, les principes de l'apprentissage supervisé, le prétraitement des données, plusieurs algorithmes classiques de classification, les principales métriques d'évaluation ainsi que les notions de sous-apprentissage et de surapprentissage. L'objectif est d'établir un cadre conceptuel général, sans aborder à ce stade une application particulière.

1.1 Introduction à l'intelligence artificielle

L'intelligence artificielle (IA) regroupe des méthodes permettant à des systèmes informatiques d'accomplir des tâches telles que la reconnaissance de formes, la prise de décision ou l'apprentissage à partir de données. L'apprentissage automatique (Machine Learning, ML) constitue une branche de l'IA dans laquelle les modèles apprennent des relations à partir d'exemples, tandis que l'apprentissage profond (Deep Learning, DL) est un sous-domaine du ML fondé sur des réseaux de neurones comportant plusieurs couches de traitement [1], [2], [3].

Cette relation hiérarchique permet de distinguer l'IA, le ML et le DL sans les confondre. La figure 1.1 en présente une représentation synthétique [1], [3].

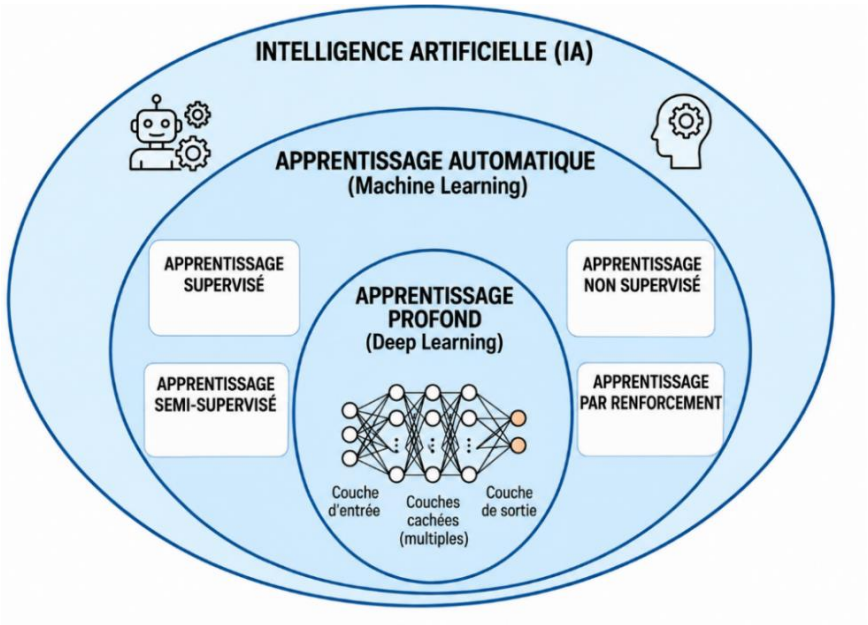

Figure 1.1– Positionnement hiérarchique de l'intelligence artificielle, de l'apprentissage automatique et de l'apprentissage profond.

Dans de nombreux domaines, l'IA est utilisée pour analyser des données, reconnaître des régularités et assister la prise de décision. L'intérêt du ML réside notamment dans sa capacité à construire un modèle à partir d'observations plutôt qu'à définir manuellement une règle distincte pour chaque situation [1], [4].

1.1.1 Notions essentielles sur l'apprentissage profond

L'apprentissage profond est une famille de méthodes d'apprentissage automatique reposant sur des réseaux de neurones organisés en plusieurs couches de traitement. Ces couches transforment progressivement les données d'entrée afin d'apprendre des représentations de niveaux d'abstraction croissants. L'ajustement des paramètres du réseau est généralement réalisé par rétropropagation de l'erreur associée à une fonction objectif [3].

Une caractéristique importante du Deep Learning est sa capacité à apprendre automatiquement des représentations utiles à partir des données, ce qui peut réduire le recours à une construction manuelle détaillée des caractéristiques. Cette flexibilité s'accompagne toutefois d'une complexité de calcul plus élevée et, selon la tâche et l'architecture considérées, d'un besoin important en données d'apprentissage. Dans ce chapitre, cette notion est volontairement limitée à son positionnement théorique au sein du Machine Learning ; les architectures profondes ne sont pas développées davantage [3].

1.2 Principes généraux et principales approches d'apprentissage automatique

L'apprentissage automatique est une branche de l'IA dans laquelle des algorithmes exploitent des données afin d'identifier des régularités et de construire des modèles capables de produire des prédictions ou des décisions [1], [4]. Le principe général consiste à représenter les observations par des variables exploitables, appelées caractéristiques (features), puis à ajuster un modèle à partir de ces informations [4].

Une chaîne générale d'apprentissage automatique comprend l'acquisition ou la constitution des données, leur prétraitement, la représentation des caractéristiques, l'entraînement du modèle, puis son utilisation pour produire une prédiction sur de nouvelles observations [1], [4]. Cette organisation est résumée à la figure 1.2.



Figure 1.2– Chaîne générale d'un système d'apprentissage automatique, de l'acquisition des données à la prédiction.

L'apprentissage automatique est généralement présenté selon quatre grandes catégories : l'apprentissage supervisé, l'apprentissage non supervisé, l'apprentissage semi-supervisé et l'apprentissage par renforcement [1], [4]. La figure 1.3 en propose une représentation synthétique.

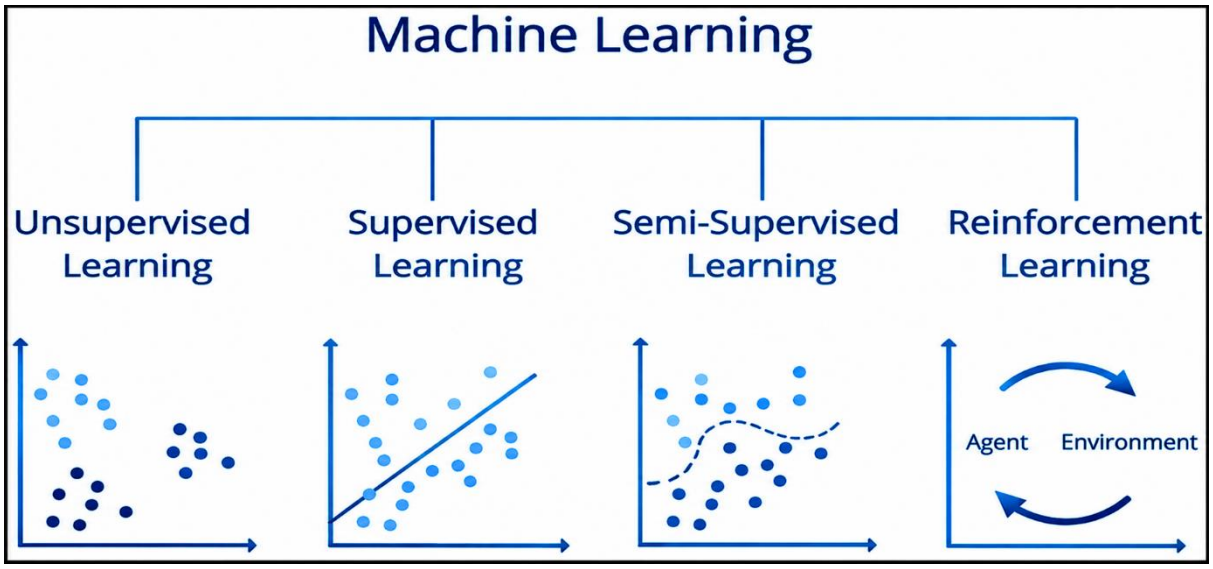

Figure 1.3– Principales catégories d'apprentissage automatique.

Afin de distinguer clairement ces quatre approches, le tableau 1.1 synthétise leur principe, le type de données utilisé et leur objectif général [1], [4].

Tableau 1-1– Comparaison des principales catégories d'apprentissage automatique.

Catégorie	**Principe de fonctionnement**	**Type de données utilisées**	**Objectif principal**	**Exemples d'applications**
Apprentissage supervisé	Le modèle apprend une relation entre des variables d'entrée et une **sortie connue**, appelée cible ou étiquette. L'apprentissage consiste à ajuster les paramètres du modèle afin de minimiser l'écart entre les prédictions et les valeurs réelles.	Données **étiquetées** : chaque observation possède des variables d'entrée et une sortie connue.	Prédire la classe ou la valeur d'une nouvelle observation à partir d'exemples connus.	Classification et régression.
Apprentissage non supervisé	Le modèle analyse les données **sans disposer d'étiquettes**	Données **non étiquetées**.	Découvrir l'organisation intrinsèque des données et	Regroupement, réduction de dimension,

	préalables afin d'identifier automatiquement des structures, des regroupements, des relations ou des observations atypiques.		identifier des structures inconnues.	exploration de données.
Apprentissage semi-supervisé	Cette approche exploite simultanément un **petit ensemble de données étiquetées** et un ensemble généralement plus important de données non étiquetées. Les données non étiquetées sont utilisées pour améliorer l'apprentissage lorsque l'annotation complète est difficile ou coûteuse.	Données **mixtes** : étiquetées et non étiquetées.	Améliorer les performances d'un modèle en limitant le besoin d'étiquetage manuel.	Classification lorsque peu de données sont étiquetées.
Apprentissage par renforcement	Un **agent** interagit avec un environnement en réalisant des actions. En fonction des conséquences de ces actions, il reçoit des récompenses ou pénalités et apprend progressivement une stratégie permettant de maximiser la récompense cumulée.	Données issues de l'**interaction agent–environnement** : états, actions, récompenses et transitions.	Apprendre une politique de décision optimale dans un problème séquentiel.	Commande, navigation et prise de décision séquentielle.

Parmi ces approches, l'apprentissage supervisé constitue le cadre de référence lorsque chaque observation est associée à une sortie connue [1], [4]. Il est détaillé dans la section suivante.

1.3 Apprentissage supervisé : principes et tâches

1.3.1 Principe de l'apprentissage supervisé

Dans l'apprentissage supervisé, le modèle est entraîné à partir d'observations étiquetées : chaque exemple associe un ensemble de variables d'entrée à une sortie connue. L'objectif est d'apprendre une relation permettant de produire une sortie pour de nouvelles observations qui n'ont pas servi à l'entraînement [1], [4].

Deux tâches principales relèvent de cette approche : la régression, qui vise à estimer une variable numérique continue, et la classification, qui attribue une observation à l'une des classes définies à l'avance [1], [4].

1.3.2 Algorithmes de régression

Les algorithmes de régression sont utilisés lorsque la variable à prédire est numérique et continue. Ils modélisent la relation entre une ou plusieurs variables d'entrée et une variable de sortie afin d'en estimer la valeur pour de nouvelles observations [1], [4].

La régression linéaire : simple ou multiple, représente la sortie comme une combinaison linéaire d'une ou de plusieurs variables explicatives. Les coefficients du modèle quantifient la contribution associée aux variables dans cette représentation [4].

La régression polynomiale : enrichit une représentation linéaire par l'introduction de termes polynomiaux afin de décrire certaines relations courbes entre les variables [4].

Plus généralement, une régression non linéaire peut être utilisée lorsque la relation étudiée ne peut pas être décrite de manière satisfaisante par une forme linéaire simple [4].

Cette distinction entre régression et classification permet d'orienter le choix de la famille de modèles en fonction de la nature de la variable de sortie [1], [4].

1.4 Prétraitement des données

La qualité des données influence directement la fiabilité de l'apprentissage. Le prétraitement regroupe les opérations destinées à rendre les observations cohérentes et exploitables avant l'entraînement d'un modèle [5].

1.4.1 Nettoyage des données

Le nettoyage des données consiste à identifier les valeurs manquantes, les doublons, les incohérences, les erreurs de saisie et les observations potentiellement aberrantes. Leur traitement dépend de leur origine : elles peuvent être corrigées, supprimées ou conservées lorsqu'elles correspondent à une observation valide. Toute suppression doit être justifiée afin de ne pas éliminer une variabilité réelle [5].

1.4.2 Normalisation et standardisation

Certains algorithmes fondés sur des distances ou sur des marges sont sensibles à l'échelle des variables. Une mise à l'échelle est alors recommandée afin d'éviter qu'une variable de grande amplitude influence artificiellement davantage le calcul [5].

La normalisation Min-Max transforme les valeurs d'une variable dans un intervalle défini, généralement [0, 1] :

$$x' = (x - x_min) / (x_max - x_min) \quad (1.1)$$

où x est la valeur initiale, et x_min et x_max représentent respectivement les valeurs minimale et maximale de la variable [5].

La standardisation centre les données autour de leur moyenne et les exprime en fonction de leur écart-type :

$$z = (x - \mu) / \sigma \quad (1.2)$$

où μ représente la moyenne et σ l'écart-type de la variable [5].

Le choix entre normalisation et standardisation dépend de l'algorithme et des propriétés des variables. Les paramètres de transformation doivent être estimés uniquement à partir des données d'apprentissage, puis appliqués sans recalcul aux ensembles de validation et de test afin d'éviter une fuite d'information [5], [6].

1.4.3 Encodage des variables catégorielles

Les variables catégorielles doivent souvent être converties en une représentation numérique. L'encodage ordinal convient lorsque les catégories possèdent un ordre naturel, tandis que l'encodage one-hot crée une variable binaire pour chaque modalité. Le choix de la méthode doit respecter la nature de la variable afin de ne pas introduire artificiellement une relation d'ordre [5].

1.4.4 Partitionnement des données et validation du modèle

Afin d'évaluer la capacité de généralisation d'un modèle, les données peuvent être séparées en ensembles d'apprentissage, de validation et de test. L'ensemble d'apprentissage sert à ajuster le modèle, l'ensemble de validation peut être utilisé pour le choix des hyperparamètres, et l'ensemble de test fournit une évaluation finale sur des observations non utilisées pour l'ajustement [7]. La figure 1.4 illustre ce principe de séparation.

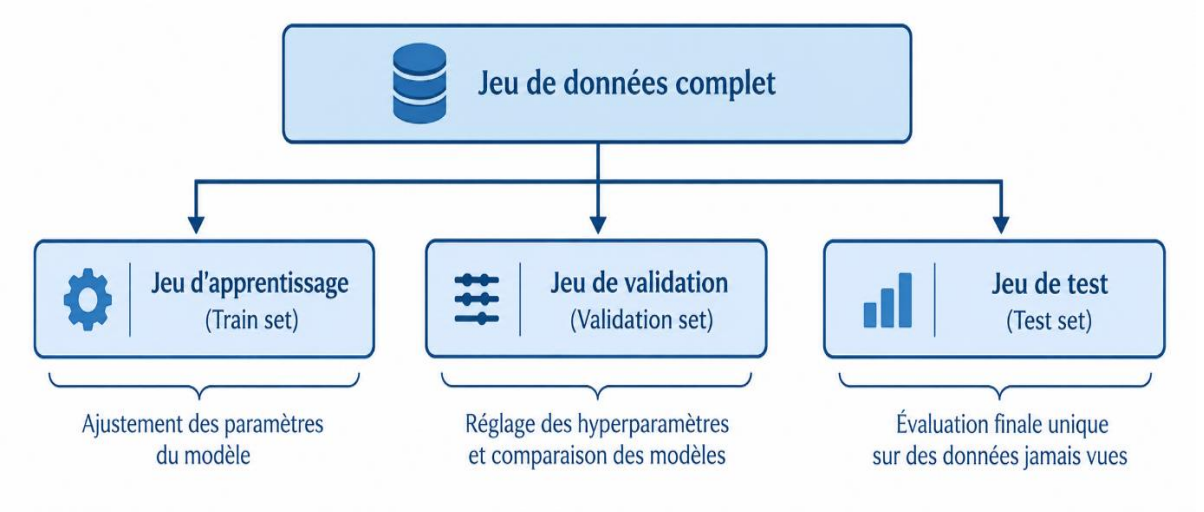

Figure 1.4– Séparation des données en ensembles d'apprentissage, de validation et de test.

La validation croisée k-fold estime les performances en répétant l'entraînement sur plusieurs partitions du jeu de données. Les observations sont réparties en k sous-ensembles ; à chaque itération, l'un d'eux sert à l'évaluation et les autres à l'apprentissage [8]. En classification, une variante stratifiée peut être utilisée afin de préserver approximativement la proportion des classes dans chaque pli [7].

Il faut également prévenir la fuite de données (data leakage), qui survient lorsqu'une information provenant des données d'évaluation influence directement ou indirectement l'apprentissage ou le choix du modèle. Cette situation peut conduire à une estimation trop optimiste des performances de généralisation [6].

1.5 Principaux algorithmes de classification

Cette section présente cinq algorithmes classiques de classification : l'arbre de décision, les k plus proches voisins, la régression logistique, la forêt aléatoire et la machine à vecteurs de support. Ils appartiennent à des familles différentes et reposent sur des mécanismes de décision distincts [1], [4], [9], [10].

1.5.1 Arbre de décision

Un arbre de décision (Decision Tree) représente une suite hiérarchique de règles. Chaque nœud interne correspond à un test appliqué à une variable, les branches représentent les résultats possibles de ce test et les feuilles portent la classe prédite [1], [4]. La figure 1.5 illustre cette organisation générale.

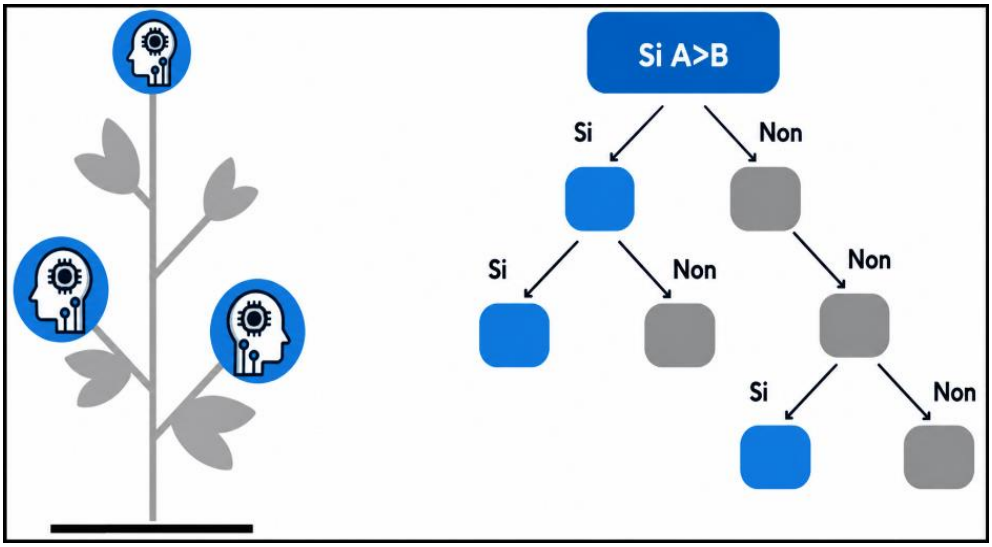

Figure 1.5– Principe d'un Arbre de décision.

La construction de l'arbre consiste à sélectionner, à chaque nœud, une règle de séparation améliorant l'homogénéité des sous-ensembles obtenus, par exemple selon l'indice de Gini ou l'entropie. La profondeur maximale, le nombre minimal d'observations requis pour diviser un nœud et la taille minimale d'une feuille font partie des hyperparamètres courants. Un arbre très complexe peut surapprendre les données, tandis qu'un arbre trop contraint peut sous-apprendre. Les arbres de décision sont interprétables et ne nécessitent généralement pas de mise à l'échelle des variables [1], [4].

1.5.2 K plus proches voisins (k-NN)

L'algorithme des k plus proches voisins (k-NN) attribue une classe à une nouvelle observation en fonction des classes représentées parmi ses k voisins les plus proches dans l'espace des caractéristiques. La proximité est définie par une métrique de distance, fréquemment la distance euclidienne [1], [4]. La figure 1.6 schématise ce principe.

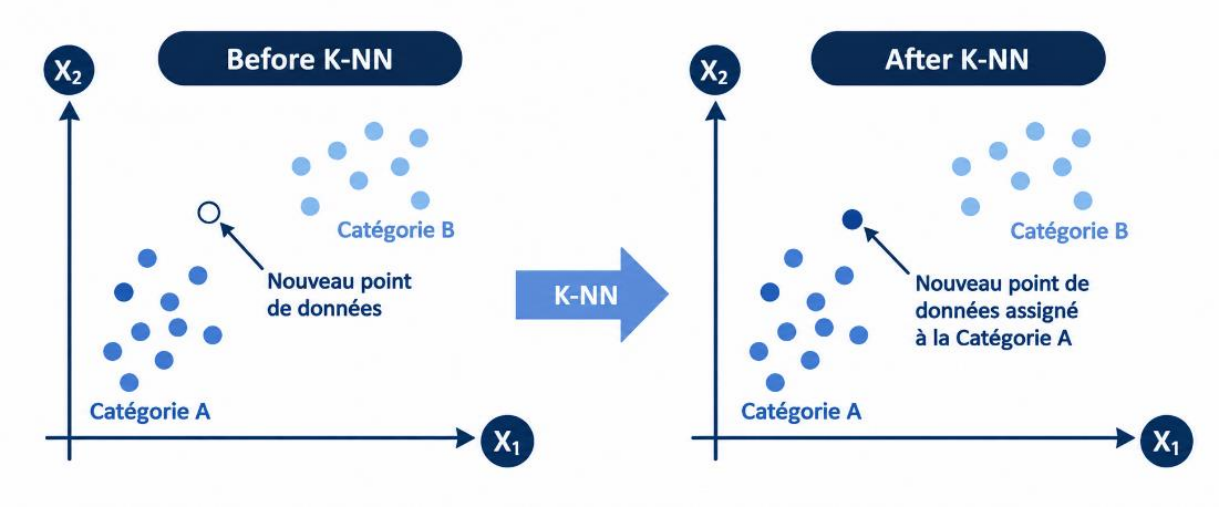

Figure 1.6– Principe du k-NN : classification par vote majoritaire des k plus proches voisins.

Le paramètre principal est le nombre de voisins k. Une valeur faible rend la décision plus sensible à la structure locale et au bruit, tandis qu'une valeur plus élevée produit une frontière de décision plus lissée. Comme la sélection des voisins repose sur une métrique de distance, la mise à l'échelle des variables peut modifier les voisinages obtenus et doit être prise en compte lors du prétraitement [4], [11]. Le k-NN conserve les observations de référence et effectue l'essentiel du calcul au moment de la prédiction [4].

1.5.3 Régression logistique

Malgré son nom, la régression logistique est une méthode de classification. Dans le cas binaire, elle estime une probabilité d'appartenance à une classe à partir d'une combinaison linéaire des variables d'entrée, transformée par la fonction logistique ou sigmoïde [1], [4]. La figure 1.7 représente la forme générale de cette fonction.

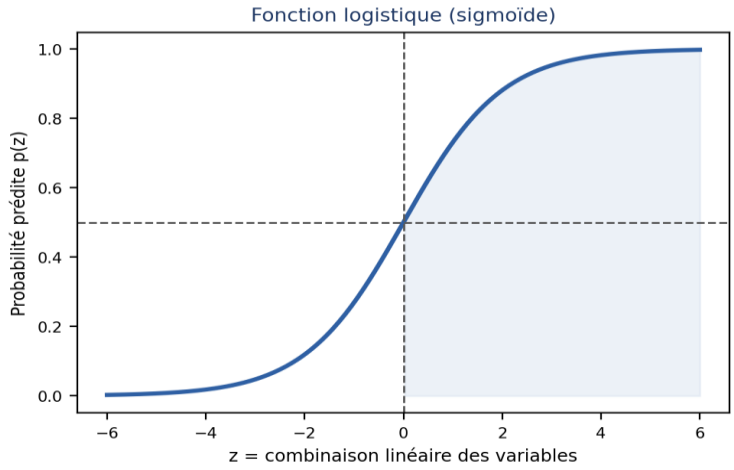

Figure 1.7– Fonction logistique utilisée pour convertir une combinaison linéaire de variables en probabilité

Dans le cas binaire, la classe prédite est obtenue en comparant la probabilité estimée à un seuil de décision. Pour plusieurs classes, une formulation multinomiale ou des stratégies de décomposition peuvent être utilisées. Une régularisation L1 ou L2 peut limiter la complexité du modèle. La régression logistique est relativement simple à entraîner et à interpréter ; sa frontière de décision est linéaire dans l'espace des variables représentées, sauf transformation de celles-ci [1], [4].

1.5.4 Forêt aléatoire (Random Forest)

La forêt aléatoire (Random Forest) est une méthode d'ensemble fondée sur plusieurs arbres de décision. Chaque arbre est entraîné à partir d'un échantillon bootstrap et, lors de la construction des arbres, un sous-ensemble aléatoire de variables est considéré à chaque division. Cette randomisation réduit la dépendance entre les arbres [9]. La figure 1.8 résume ce mécanisme.

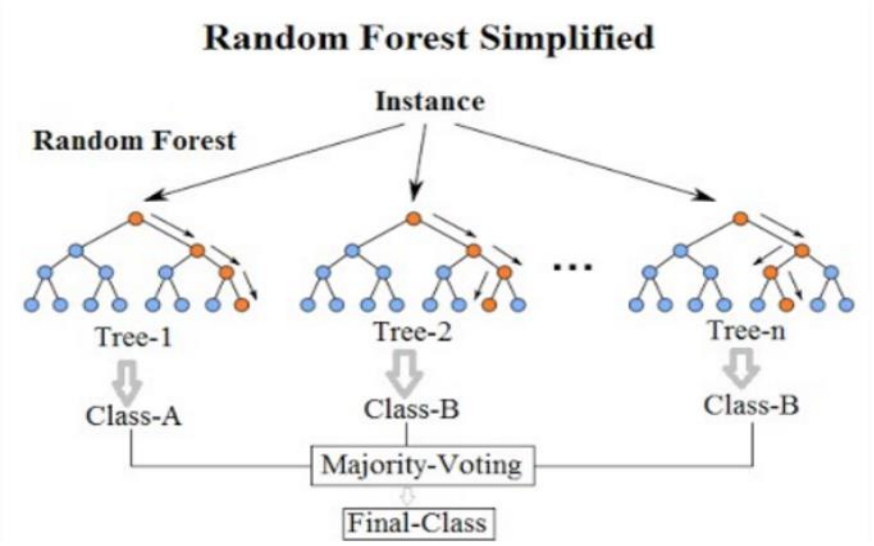

Figure 1.8– Principe de la forêt aléatoire : agrégation des décisions de plusieurs arbres entraînés sur des échantillons bootstrap.

En classification, la prédiction finale d'une forêt aléatoire est généralement obtenue par agrégation des décisions des arbres. Cette combinaison réduit la variance par rapport à un arbre isolé. Les hyperparamètres courants comprennent notamment le nombre d'arbres, leur profondeur et le nombre de variables candidates à chaque division. La méthode ne requiert généralement pas de normalisation des variables et permet d'estimer leur importance selon différentes mesures [9].

1.5.5 Machine à vecteurs de support (SVM)

La machine à vecteurs de support (Support Vector Machine, SVM) recherche un hyperplan séparateur qui maximise la marge entre les classes. Les observations les plus proches de la frontière, appelées vecteurs de support, déterminent directement la solution [10].

Lorsque la séparation linéaire est insuffisante, une fonction noyau peut être utilisée pour construire une séparation non linéaire dans l'espace d'origine. Le paramètre C contrôle le compromis entre la largeur de la marge et les erreurs de classification ; pour un noyau radial (RBF), le paramètre gamma règle l'étendue de l'influence des observations [10]. La mise à l'échelle des caractéristiques peut modifier la géométrie du problème et donc la frontière obtenue par un SVM ; une standardisation est couramment utilisée, mais son effet doit être évalué selon les données considérées [12].

La figure 1.9 illustre la notion de marge maximale.

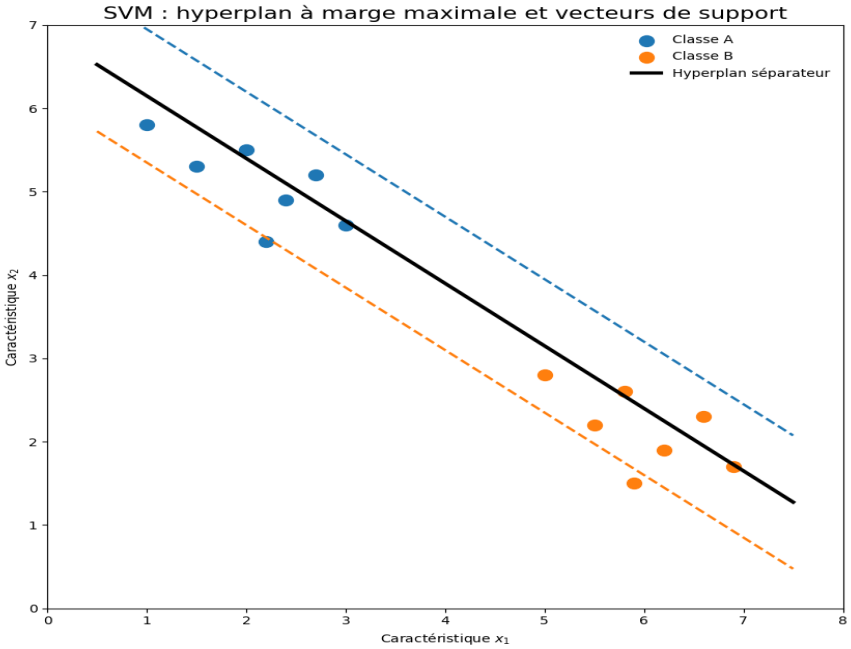

Figure 1.9– Principe du SVM : hyperplan à marge maximale et vecteurs de support.

Le tableau 1.2 regroupe les caractéristiques essentielles des cinq algorithmes présentés afin de faciliter leur comparaison théorique [1], [4], [9], [10].

Tableau 1-2– Synthèse comparative de cinq algorithmes classiques de classification.

Algorithme	Prétraitement requis	Hyperparamètres clés	Avantages	Limites
Arbre de décision	Mise à l'échelle généralement non requise	Profondeur max., taille min. de feuille	Interprétable, règles explicites	Peut surapprendre si sa complexité est mal contrôlée
k-NN	Mise à l'échelle généralement nécessaire	Nombre de voisins k, métrique de distance	Simple, non paramétrique	Sensible à l'échelle et coûteux en prédiction sur de grands jeux
Régression logistique	Mise à l'échelle souvent recommandée	Régularisation (L1/L2)	Simple, probabiliste, interprétable	Frontière de décision linéaire dans l'espace des variables
Forêt aléatoire	Mise à l'échelle généralement non requise	Nombre d'arbres, profondeur, variables par nœud	Réduit la variance, importance des variables possible	Moins interprétable qu'un arbre unique ; coût accru avec le nombre d'arbres
SVM	Standardisation généralement recommandée	C, noyau, gamma	Frontières linéaires ou non linéaires via noyaux	Réglage sensible du noyau et des hyperparamètres

Les performances relatives de ces algorithmes dépendent notamment de la structure des données, du nombre d'observations, du prétraitement et du réglage des hyperparamètres. Leur comparaison doit donc reposer sur un protocole d'évaluation commun et appliqué de manière cohérente [4], [8].

1.6 Critères d'évaluation

1.6.1 Matrice de confusion

Pour un problème de classification binaire, la matrice de confusion distingue quatre situations : le vrai positif (VP), le vrai négatif (VN), le faux positif (FP) et le faux négatif (FN). Elle permet de relier les classes réelles aux classes prédites et constitue la base de plusieurs métriques d'évaluation [13]. La figure 1.10 en présente la structure générale.

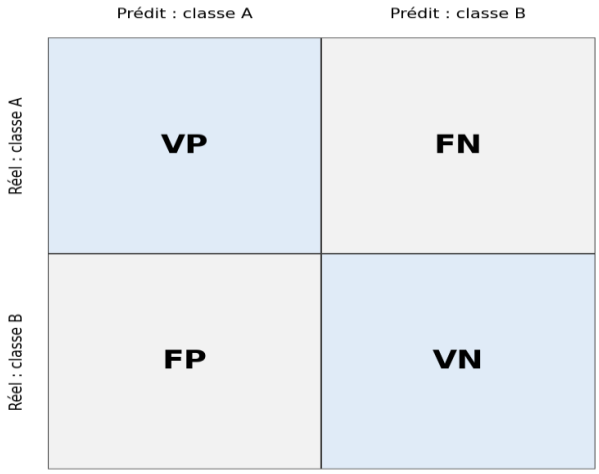

Figure 1.10– Structure générale d'une matrice de confusion binaire.

Dans un problème multiclasse, la matrice de confusion est étendue à autant de lignes et de colonnes qu'il existe de classes. Elle permet d'identifier les classes correctement prédites ainsi que les confusions entre classes, information qu'un indicateur global ne fournit pas à lui seul [13].

1.6.2 Accuracy, Precision et Recall

À partir de la matrice de confusion, plusieurs métriques complémentaires peuvent être calculées. L'exactitude (accuracy) correspond à la proportion globale de prédictions correctes [13] :

$$\text{Accuracy} = (VP + VN) / (VP + VN + FP + FN) \quad (1.3)$$

La précision (precision) correspond à la proportion de prédictions positives qui sont effectivement positives [13] :

$$\text{Precision} = VP / (VP + FP) \quad (1.4)$$

Le rappel (recall), ou sensibilité, correspond à la proportion des observations réellement positives correctement identifiées [13] :

$$\text{Recall} = VP / (VP + FN) \quad (1.5)$$

L'accuracy peut être trompeuse lorsque les classes sont fortement déséquilibrées, car un modèle privilégiant la classe majoritaire peut conserver un score global élevé tout en offrant de faibles performances sur les classes minoritaires [13]. Il est donc préférable d'interpréter conjointement plusieurs métriques.

Dans un problème multiclasse, la précision, le rappel et le F1-score peuvent être calculés pour chaque classe, puis agrégés. La moyenne macro attribue le même poids à chaque classe, la

moyenne micro agrège les décisions de l'ensemble des classes, et la moyenne pondérée tient compte de l'effectif de chaque classe [13].

1.6.3 F1-score

Le F1-score est la moyenne harmonique de la précision et du rappel [13] :

$$F1 = 2 \times (Precision \times Recall) / (Precision + Recall) \quad (1.6)$$

Comme il s'agit d'une moyenne harmonique, le F1-score diminue lorsque l'une des deux composantes est faible. Il fournit ainsi un indicateur synthétique lorsqu'il est nécessaire de considérer simultanément la précision et le rappel [13].

Le tableau 1.3 résume les principales métriques présentées, leur expression et leur limite d'interprétation [13].

Tableau 1-3 Synthèse des principales métriques de classification.

Métrique	Formule	Ce qu'elle mesure	Limite principale
Accuracy	(VP+VN) /(VP+VN+FP+FN)	Proportion globale de bonnes prédictions	Peut masquer de faibles performances sur une classe minoritaire
Precision	VP/(VP+FP)	Proportion de prédictions positives correctes	Ne mesure pas les positifs manqués
Recall	VP/(VP+FN)	Proportion de positifs réels correctement identifiés	Ne mesure pas les faux positifs
F1-score	2·P·R/(P+R)	Compromis Precision / Recall	Moins intuitif à interpréter isolément

1.7 Surapprentissage et sous-apprentissage

1.7.1 Définitions et causes

Le sous-apprentissage (underfitting) apparaît lorsqu'un modèle est trop simple pour représenter correctement la structure des données ; ses performances sont alors insuffisantes sur les données d'apprentissage comme sur des données nouvelles. À l'inverse, le surapprentissage (overfitting) se produit lorsqu'un modèle s'adapte excessivement aux particularités du jeu d'apprentissage et généralise mal à de nouvelles observations [14].

1.7.2 Compromis biais-variance

Le compromis biais-variance décrit la relation entre deux composantes de l'erreur de généralisation. Un biais élevé est généralement associé à une représentation trop simple, tandis qu'une variance élevée traduit une forte sensibilité du modèle aux fluctuations du jeu d'apprentissage. La complexité du modèle doit être ajustée afin d'obtenir un compromis favorable entre ces deux composantes [15]. La figure 1.11 illustre ce principe de manière conceptuelle.

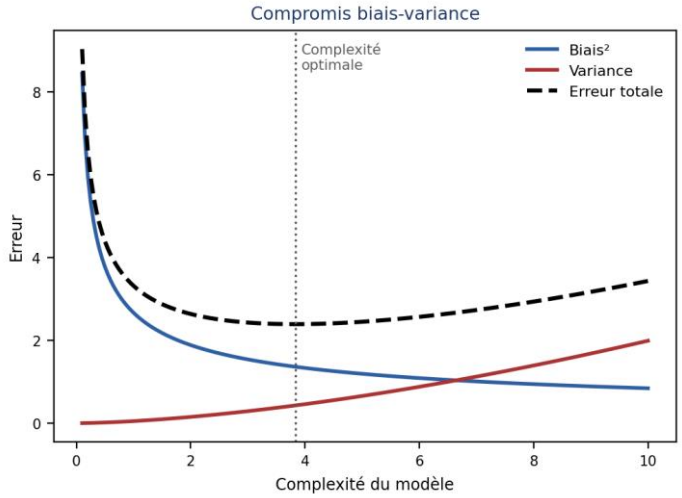

Figure 1.11– Illustration conceptuelle du compromis biais-variance en fonction de la complexité du modèle

1.7.3 Moyens de détection et de mitigation

Le surapprentissage peut être détecté en comparant les performances obtenues sur les données d'apprentissage et sur des données indépendantes de l'ajustement. La validation croisée, la régularisation, la maîtrise de la complexité du modèle et le réglage des hyperparamètres sur des données distinctes du test final figurent parmi les stratégies courantes pour limiter ce phénomène [8], [14], [15].

Conclusion

Ce chapitre a présenté les principaux fondements théoriques de l'apprentissage automatique utiles à la compréhension des problèmes de classification : positionnement de l'IA, du ML et du DL, principales catégories d'apprentissage, apprentissage supervisé, prétraitement des données, algorithmes classiques de classification, métriques d'évaluation et notions de

sous-apprentissage et de surapprentissage. Ces éléments constituent le cadre conceptuel nécessaire pour aborder l'état de l'art du domaine d'application dans le chapitre suivant.

Chapitre 2

Intelligence artificielle appliquée à la détection des fuites d’eau : état de l’art

Introduction

La détection des fuites constitue un enjeu important pour la surveillance des réseaux et des installations hydrauliques. Une fuite peut provoquer des variations de pression, de débit, de bruit ou de vibration, mais ces variations ne sont pas toujours spécifiques à un défaut : elles peuvent également résulter des conditions normales d'exploitation, de variations de consommation ou de modifications du régime hydraulique [16], [17].

Les approches conventionnelles reposent principalement sur l'analyse hydraulique, les méthodes acoustiques et vibratoires ainsi que sur la comparaison entre mesures réelles et modèles hydrauliques. Le développement des systèmes d'acquisition et l'augmentation des données disponibles ont favorisé l'utilisation de méthodes fondées sur l'intelligence artificielle et l'apprentissage automatique pour automatiser la détection, la classification ou la localisation des anomalies [17], [18].

Ce chapitre présente les principes essentiels nécessaires pour situer le présent travail : principales familles de méthodes, types de données exploités et travaux récents utilisant l'intelligence artificielle. L'objectif est de dégager les tendances, les apports et les limites les plus directement liés à la classification d'états de fonctionnement à partir de mesures hydrauliques.

2.1 Des méthodes conventionnelles aux approches basées sur les données

2.1.1 Méthodes conventionnelles de détection

Les techniques conventionnelles de détection peuvent être regroupées en trois familles principales. Les méthodes hydrauliques analysent les variations de pression, de débit ou de bilan d'eau. Les méthodes acoustiques et vibratoires recherchent les signaux produits par l'écoulement au niveau d'une fuite. Enfin, les méthodes basées sur des modèles comparent les grandeurs mesurées aux valeurs calculées par un modèle hydraulique [16], [17].

Chaque famille possède des limites. Une variation hydraulique peut être provoquée par une modification normale du fonctionnement ; les signaux acoustiques sont sensibles au bruit et aux caractéristiques des conduites ; les méthodes basées sur un modèle dépendent de la qualité de la modélisation et de sa calibration. Ces contraintes expliquent l'intérêt croissant des approches basées sur les données, capables d'exploiter simultanément plusieurs variables afin d'identifier des relations plus complexes qu'un simple seuil appliqué à une grandeur isolée [17], [18].

Le tableau 2.1 résume ces trois familles. Il sert uniquement à situer l'évolution vers les approches basées sur les données ; les méthodes conventionnelles ne sont donc pas développées davantage dans ce chapitre.

Tableau 2-1 Synthèse des principales méthodes conventionnelles de détection des fuites

Méthode	Principe	Données principales	Limite principale
Hydraulique	Analyse des variations du fonctionnement hydraulique	Pression, débit	Une variation peut avoir une autre origine qu'une fuite
Acoustique / vibratoire	Détection du bruit ou des vibrations associés à la fuite	Signaux acoustiques, vibrations	Sensibilité au bruit et aux caractéristiques des conduites
Basée sur un modèle	Comparaison entre mesures réelles et valeurs calculées	Modèle hydraulique + mesures	Dépendance à la qualité et à la calibration du modèle

2.2 Rôle de l'intelligence artificielle et données exploitées

2.2.1 Tâches du diagnostic intelligent

Dans la littérature, l'expression « détection de fuites » recouvre plusieurs tâches. La détection indique si une fuite est présente ; la classification associe une observation à un état prédéfini ; la localisation recherche la zone ou la conduite concernée ; l'estimation vise, par exemple, à quantifier la sévérité ou le débit associé à la fuite [17], [18].

Ces tâches ne présentent pas le même niveau de difficulté. Un modèle peut distinguer correctement un état normal d'un état de fuite sans être capable de localiser précisément le défaut. Cette distinction est importante lors de l'interprétation des performances publiées.

2.2.2 Données hydrauliques, acoustiques et origine des observations

La pression et le débit figurent parmi les grandeurs hydrauliques les plus utilisées. Une fuite peut modifier la pression et le débit, mais ces variables restent également influencées par le fonctionnement normal du système. Leur analyse conjointe avec d'autres informations peut donc fournir une représentation plus complète de l'état hydraulique [17], [24].

Les signaux acoustiques et vibratoires constituent une autre source d'information. Ils peuvent être exploités à partir de caractéristiques extraites du signal ou de représentations telles que les spectrogrammes. Leur utilisation sur le terrain reste toutefois sensible au bruit ambiant, au

matériau des conduites, à l'atténuation du signal et au positionnement des capteurs [26], [27], [34].

Les données utilisées pour l'apprentissage proviennent principalement de simulations hydrauliques, de bancs expérimentaux ou de réseaux réels. Les simulations facilitent la génération de nombreux scénarios étiquetés, mais peuvent présenter un écart avec les conditions réelles. Les bancs expérimentaux permettent d'étudier des fuites physiques dans un environnement contrôlé, tandis que les données de réseaux réels offrent une meilleure représentativité mais restent plus difficiles à obtenir et à annoter [18]. La qualité, la quantité, la représentativité des données et la séparation correcte entre apprentissage et évaluation influencent directement la fiabilité des performances obtenues [18], [34].

2.3 État de l'art et synthèse comparative des approches récentes

Les travaux récents utilisent des méthodes allant des algorithmes classiques d'apprentissage supervisé aux architectures profondes. Les approches hydrauliques exploitent principalement la pression et le débit ; les approches acoustiques s'appuient sur le bruit et les vibrations ; les méthodes semi-supervisées cherchent à exploiter le fonctionnement normal lorsque les exemples de fuite étiquetés sont rares. Les travaux plus récents intègrent également les dépendances temporelles et la structure topologique des réseaux au moyen de Transformers ou de réseaux neuronaux sur graphes [19] – [41].

Afin d'éviter une présentation répétitive de chaque publication, le tableau 2.2 regroupe les principaux travaux retenus selon des critères communs : données, méthode d'intelligence artificielle, tâche visée, résultat principal rapporté et limite de l'étude. Les résultats sont présentés tels qu'annoncés par les auteurs et ne constituent pas un classement direct, car les bases, réseaux et protocoles d'évaluation diffèrent d'une étude à l'autre [17], [18].

Tableau 2-2 Synthèse comparative des travaux récents sur la détection des fuites d'eau par intelligence artificielle.

Étude	Données / contexte	Méthode d'IA	Tâche	Résultat principal rapporté	Limite / portée
Sun et al. [19], 2020	Pressions ; réseau Hanoi simulé	Krigeage + LDA / ANN + mise à jour bayésienne	Localisation	> 80 % dans les meilleurs scénarios rapportés	Validation simulée ; comportement sur réseau réel non démontré
Taghlabi et al. [21], 2020	EPANET/MATLAB + essais sur hydrants à Casablanca	Random Forest + modèle hydraulique	Détection / prélocalisation	Prélocalisation dans un rayon d'environ 100 m pour les essais présentés	Dépendance au réseau étudié et à la calibration du modèle

Étude	Données / contexte	Méthode d'IA	Tâche	Résultat principal rapporté	Limite / portée
Mashhadi et al. [24], 2021	Pression + débit ; campus de Lille simulé avec EPANET	LR, DT, RF, ANN + méthodes non supervisées	Détection / localisation	Avantage global des méthodes supervisées dans l'étude	Données principalement simulées
Lučin et al. [25], 2021	Scénarios Monte-Carlo ; variations de demande, position et intensité de fuite	Random Forest	Localisation	Amélioration lorsque plusieurs nœuds candidats sont considérés	Sensibilité aux faibles fuites et aux incertitudes sur la demande
Vanijjirattikhan et al. [26], 2022	Signaux acoustiques provenant de réseaux exploités	SVM, CNN, DNN	Détection	DNN ≈ 94,9 % en test ; > 90 % lors des essais terrain rapportés	Dépendance aux conditions acoustiques étudiées
Tijani et al. [27], 2022	Données réelles issues d'enregistreurs de bruit	Sélection de caractéristiques + ML	Détection	Les caractéristiques sélectionnées améliorent les résultats par rapport aux caractéristiques brutes	Sensibilité au bruit et à la qualité du signal
Tornyeviadzi et al. [30], 2023	LeakDB	Dix méthodes semi-supervisées	Détection d'anomalies	Local Outlier Factor : meilleur compromis global selon les critères de l'étude	Benchmark généré ; représentativité opérationnelle limitée
Sousa et al. [31], 2023	Pressions réelles ; secteur résidentiel de Stockholm	K-means + Learning Vector Quantization	Classification	Jusqu'à 93,98 % de classifications correctes	Jeu de données non public ; reproductibilité limitée
Rajabi et al. [32], 2023	Pression + demande hydraulique	Conditional Deep Convolutional GAN	Détection / localisation	Détection fondée sur l'écart entre l'état observé et l'état normal appris	Dépendance à la qualité du modèle représentant l'état normal
Wu et al. [34], 2024	Signaux acoustiques réels	Augmentation de données + CNN	Détection	IAAFT : gain > 7 points d'exactitude dans l'expérience rapportée	Généralisation entre réseaux à confirmer
Luo et al. [35], 2024	Séries temporelles de pression ; trois réseaux simulés	Transformer	Détection	Performances supérieures au CNN de référence dans l'étude	Validation principalement simulée

Étude	Données / contexte	Méthode d'IA	Tâche	Résultat principal rapporté	Limite / portée
Jun et al. [36], 2024	Infrastructure réelle d'Austin ; fuites simulées	CNN 3D	Détection / localisation	Performances supérieures à la méthode de référence utilisée	Dépendance au modèle hydraulique ; réentraînement parfois nécessaire
Li et Wu [37], 2024	Deux réseaux urbains ; simulation + événements réels	Convolutional GNN	Localisation	90,28 % d'exactitude dans la configuration à forte densité de mesures	Dépendance au nombre et au placement des capteurs
Li et al. [39], 2025	Pression, demande et topologie du réseau	Graph Convolutional Network	Détection / localisation	Construction d'une carte des zones les plus probables	Dépendance à la représentation topologique et aux données disponibles
Zhang et Fink [40], 2026	Données hydrauliques + structure du réseau	GNN informé par un algorithme de flot	Détection / localisation	Approche visant à améliorer la généralisation lorsque les conditions changent	Complexité accrue et dépendance à la représentation du réseau
Ranganathan et al. [41], 2026	> 47 000 scénarios Net3	RF, XGBoost, LightGBM, ANN, CNN, LSTM, Transformer, GNN	Détection / localisation	> 95 % en détection binaire ; Top-1 ≈ 75–77 % pour les meilleurs modèles d'ensemble en localisation	Benchmark simulé ; transfert au terrain non démontré

Source : synthèse de l'auteure à partir de [19], [21], [24] – [27], [30] – [32], [34] –[37], [39] –[41].

La comparaison met en évidence plusieurs constats. Premièrement, les données de pression et de débit restent largement utilisées dans les approches hydrauliques. Deuxièmement, les données réelles sont moins fréquentes que les données simulées, principalement en raison de la difficulté à disposer d'événements de fuite correctement identifiés. Troisièmement, les modèles les plus complexes ne garantissent pas à eux seuls une meilleure généralisation : leurs performances restent fortement liées à la représentativité des données et au protocole de validation.

Enfin, la détection ou la classification d'un état apparaît généralement moins exigeante que la localisation précise d'une fuite sur un réseau étendu. Les valeurs de performance doivent donc toujours être interprétées en tenant compte de la tâche réalisée, de l'origine des données et des conditions de validation.

2.4 Positionnement du présent travail

Le travail développé dans ce mémoire porte sur la classification supervisée de plusieurs états de fonctionnement à partir de données issues d'une maquette hydraulique instrumentée. Les variables exploitées sont principalement le débit, la pression, le niveau du réservoir et l'état de fonctionnement de la pompe.

L'objectif n'est pas de localiser une fuite sur un grand réseau urbain, mais d'étudier si l'analyse conjointe de plusieurs grandeurs mesurées permet de distinguer automatiquement les états définis lors des essais. Cette orientation est cohérente avec les travaux montrant l'intérêt des approches multivariées fondées sur des données hydrauliques [17], [24].

L'utilisation d'une base expérimentale permet de travailler avec des observations directement issues du prototype étudié. Les résultats restent néanmoins liés aux conditions d'essai, aux classes disponibles et aux plages de fonctionnement représentées dans cette base. Ils ne peuvent donc pas être généralisés directement à un réseau de distribution réel sans validation complémentaire.

Le chapitre suivant présente la structure de la base expérimentale, les étapes de préparation des données, le protocole d'apprentissage et l'évaluation du modèle de classification.

Conclusion

Ce chapitre a présenté les principales méthodes de détection des fuites et leur évolution vers des approches fondées sur l'intelligence artificielle. Les travaux étudiés exploitent principalement des données hydrauliques, acoustiques ou spatio-temporelles pour réaliser des tâches de détection, de classification ou de localisation.

L'analyse de la littérature montre que la performance d'un modèle dépend non seulement de l'algorithme utilisé, mais également de la qualité et de la représentativité des données ainsi que du protocole de validation. L'écart entre les données simulées et les conditions réelles constitue également une limite récurrente.

Dans ce contexte, le présent travail s'inscrit dans une approche expérimentale de classification supervisée fondée sur plusieurs grandeurs hydrauliques. Cette orientation permet d'évaluer la capacité d'un modèle d'apprentissage à distinguer automatiquement différents états de fonctionnement à partir des mesures obtenues sur le prototype.

Chapitre 3

Préparation des données, apprentissage et validation du système de classification

Introduction

Ce chapitre présente la mise en œuvre de la partie apprentissage automatique à partir de la base de données obtenue lors des essais du prototype. Il décrit successivement la vérification et l'analyse des données, leur préparation pour l'apprentissage, la construction et l'optimisation des classifieurs, puis l'évaluation du modèle retenu.

La dernière partie est consacrée à l'intégration du modèle dans la plateforme web de supervision. Des scénarios issus de la base expérimentale sont réinjectés via un script Python exécuté sous VS Code afin de vérifier l'ensemble de la chaîne logicielle, depuis la réception des données par le serveur Flask jusqu'à la prédiction de l'état du système et son affichage sur le tableau de bord. Les résultats obtenus sont ensuite analysés en tenant compte des limites du protocole de validation.

3.1 Démarche générale du travail

La démarche mise en œuvre s'articule en sept étapes successives, résumées au Tableau 3.1. Cette organisation distingue clairement le travail de préparation des données, le travail de modélisation et le travail de validation, et constitue le fil conducteur de l'ensemble du chapitre.

Tableau 3-1–Organisation générale de la démarche expérimentale et de modélisation

Étape	Travail effectué	Finalité
1	Chargement et inspection de la base	Identifier dimensions, types, valeurs manquantes et cohérence générale
2	Nettoyage et encodage	Obtenir des variables numériques exploitables
3	Analyse exploratoire	Étudier distributions, dispersion, corrélations et déséquilibre des classes
4	Équilibrage et standardisation	Préparer les entrées pour l'apprentissage
5	Modèle de référence	Établir un premier niveau de performance avec une Random Forest
6	Optimisation Optuna	Comparer plusieurs familles de modèles et rechercher automatiquement les hyperparamètres
7	Évaluation, sauvegarde et vérification	Analyser les erreurs et préparer le modèle pour une utilisation ultérieure

3.2 Base de données expérimentale

La première étape consiste à caractériser la base issue des essais avant toute opération d'apprentissage. Cette vérification porte sur sa structure, la qualité des enregistrements et la répartition des états à classifier.

3.2.1 Structure des données

La base initiale contient 895 enregistrements et cinq variables. Le débit, la pression, le niveau du réservoir et l'état de la pompe constituent les variables d'entrée ; la variable etat_systeme représente la classe à prédire. Le problème étudié est ainsi formulé comme une classification supervisée multiclasse comportant huit états de fonctionnement.

Tableau 3-2– Variables utilisées pour la classification.

Variable	Nature	Rôle	Unité / modalité
Debit_L_min	Numérique	Entrée	L/min
Pression_bar	Numérique	Entrée	Bar
Niveau_pct	Numérique	Entrée	%
Etat_pompe	Catégorielle	Entrée	ACTIVE / INACTIVE
Etat_systeme	Catégorielle	Cible	8 classes

La capture suivante présente la structure initiale de la base de données chargée dans le notebook.

```
(895, 5)
------------------------------------------------------------------------
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 895 entries, 0 to 894
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   debit_L_min   895 non-null    float64
 1   pression_bar  895 non-null    float64
 2   niveau_pct    895 non-null    float64
 3   etat_pompe    895 non-null    object
 4   etat_systeme  895 non-null    object
dtypes: float64(3), object(2)
memory usage: 35.1+ KB
------------------------------------------------------------------------
Index(['debit_L_min', 'pression_bar', 'niveau_pct', 'etat_pompe',
       'etat_systeme'],
      dtype='object')
```

Figure 3.1— Structure initiale de la base de données dans le notebook

3.2.2 Contrôle des données

La vérification du fichier ne révèle aucune valeur manquante dans les cinq variables. En revanche, 22 lignes strictement dupliquées sont identifiées puis supprimées. Cette opération

évite qu'un même enregistrement soit comptabilisé plusieurs fois lors de l'analyse et de l'apprentissage.

La base retenue pour la suite du traitement contient donc 873 observations distinctes.

Les captures suivantes illustrent les principales vérifications effectuées sur la base avant l'apprentissage.

Figure 3.2– Structure de la base après suppression des doublons.

```
# Dropping the duplicate rows
# Vérifier le nombre de lignes dupliquées avant suppression
duplicate_rows = df[df.duplicated()]
print("Number of duplicate rows before:", duplicate_rows.shape[0])

# Supprimer les doublons
df.drop_duplicates(inplace=True)

# Vérifier après suppression
duplicate_rows = df[df.duplicated()]
print("Number of duplicate rows after:", duplicate_rows.shape[0])
```

```
Number of duplicate rows before: 22
Number of duplicate rows after: 0
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 873 entries, 0 to 894
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   debit_L_min   873 non-null    float64
 1   pression_bar  873 non-null    float64
 2   niveau_pct    873 non-null    float64
 3   etat_pompe    873 non-null    int64
 4   etat_systeme  873 non-null    int64
dtypes: float64(3), int64(2)
memory usage: 40.9 KB
```



Figure 3.3– Vérification des valeurs manquantes dans la base.

3.2.3 Répartition des états de fonctionnement

Après suppression des doublons, la variable cible est répartie en huit classes correspondant aux états définis lors des essais. La distribution n'est toutefois pas homogène : les classes FUITE_IMPORTANTE, NORMAL et FUITE_FAIBLE regroupent la majorité des observations, alors que les états associés à un réservoir vide sont peu représentés.

Tableau 3-3–Effectifs des classes dans la base nettoyée.

Code	État du système	Effectif
0	FUITE_IMPORTANTE	248
1	NORMAL	227
2	FUITE_FAIBLE	218
3	FUITE_FAIBLE + RESERVOIR_PRESQUE_VIDE	53
4	FUITE_IMPORTANTE + RESERVOIR_PRESQUE_VIDE	51
5	NORMAL + RESERVOIR_PRESQUE_VIDE	50
6	NORMAL + RESERVOIR_VIDE	14
7	FUITE_FAIBLE + RESERVOIR_VIDE	12

La figure 3.4 met en évidence ce déséquilibre. Il doit être pris en compte avant l'apprentissage afin que les classes les plus fréquentes ne dominent pas la construction du modèle.

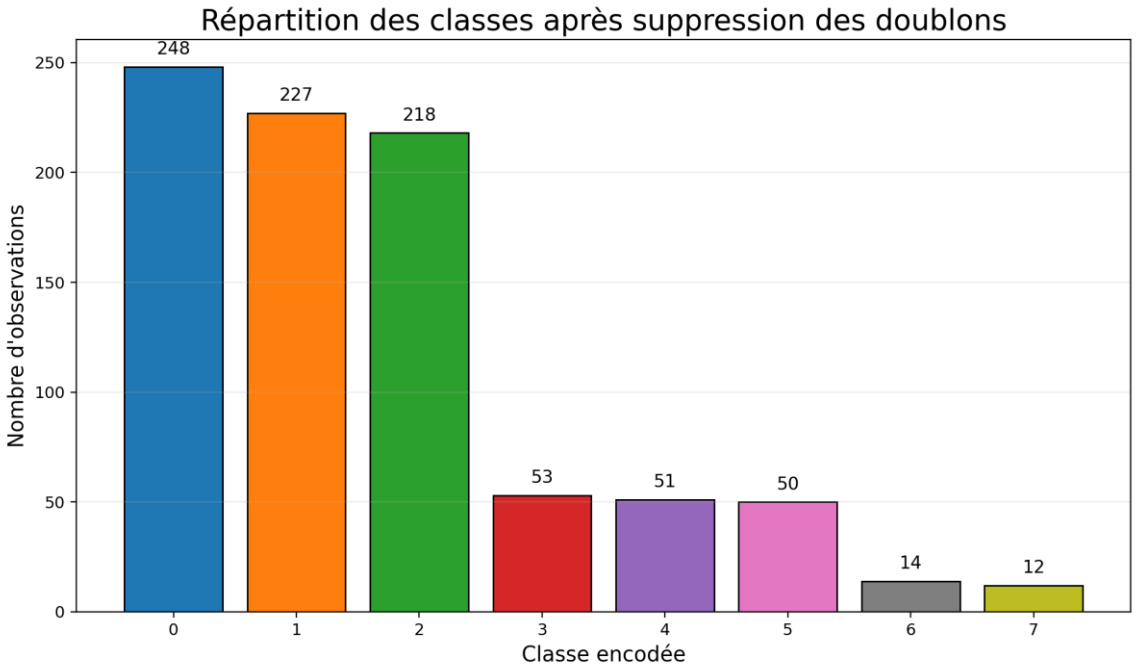

Figure 3.4– Répartition des huit classes après suppression des doublons

3.3 Analyse exploratoire des données

L'analyse exploratoire permet de décrire les mesures réellement présentes dans la base et d'examiner les relations statistiques entre les variables avant la phase d'apprentissage.

3.3.1 Statistiques descriptives

Les principales statistiques des trois grandeurs continues sont calculées sur les 873 observations retenues. Elles renseignent sur les valeurs centrales, la dispersion et les limites effectivement observées dans la base.

Tableau 3-4 – Statistiques descriptives des variables numériques

Variable	Moyenne	Écart-type	Minimum	Médiane	Maximum
Débit (L/min)	5,084	2,676	0,000	5,500	9,650
Pression (bar)	0,215	0,086	0,000	0,242	0,326
Niveau (%)	49,056	28,703	0,000	49,350	100,000

Ces valeurs décrivent uniquement la base expérimentale utilisée pour l'apprentissage et ne correspondent pas aux plages maximales des capteurs.

3.3.2 Analyse des corrélations entre les variables d'entrée

La matrice de corrélation de Pearson montre que la relation la plus importante est observée entre la pression et le niveau du réservoir ($r \approx 0{,}489$). Les corrélations débit–niveau ($r \approx 0{,}258$) et débit–pression ($r \approx 0{,}196$) sont plus faibles.

L'état de la pompe présente également des corrélations négatives avec le débit, la pression et le niveau. Globalement, ces résultats indiquent que les variables d'entrée apportent des informations complémentaires pour l'apprentissage du modèle.

La figure 3.5 présente la matrice de corrélation obtenue pour les variables d'entrée.

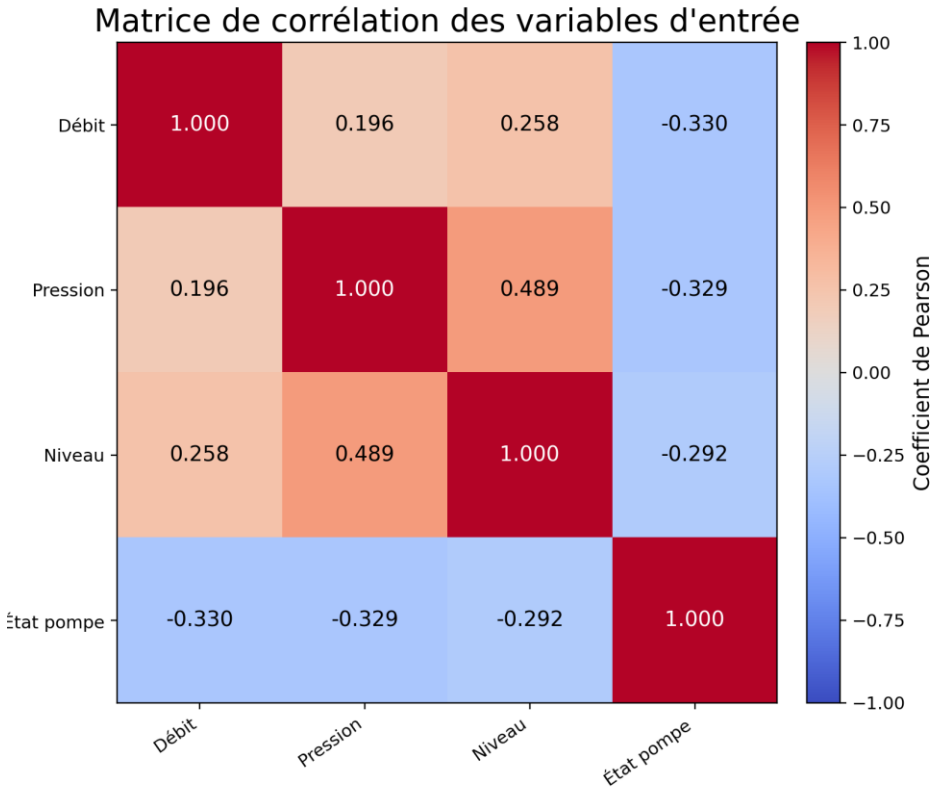

Figure 3.5– Matrice de corrélation de Pearson des variables d'entrée.

3.4 Préparation des données pour l'apprentissage

La préparation des données regroupe l'encodage des variables catégorielles, le rééquilibrage des classes et la mise à l'échelle des variables avant la séparation en ensembles d'apprentissage et de test.

3.4.1 Encodage des variables catégorielles

L'état de la pompe est converti en variable numérique selon ACTIVE = 0 et INACTIVE = 1. Les huit états de la variable cible sont également associés aux codes 0 à 7 selon l'ordre présenté au tableau 3.3 Ces codes sont utilisés comme identifiants de classe et n'impliquent aucun ordre entre les états.

3.4.2 Équilibrage des classes par SMOTE

Pour réduire le déséquilibre observé précédemment, le protocole exécuté applique SMOTE (random_state = 42). Des observations synthétiques sont générées pour les classes minoritaires jusqu'à atteindre l'effectif de la classe majoritaire, soit 248 observations par classe. Le jeu passe ainsi de 873 à 1 984 observations.

Tableau 3-5–Répartition des classes avant et après SMOTE.

Classe	Avant SMOTE	Après SMOTE
0	248	248
1	227	248
2	218	248
3	53	248
4	51	248
5	50	248
6	14	248
7	12	248

Cette étape permet d'obtenir un ensemble équilibré pour l'apprentissage. L'ordre exact dans lequel SMOTE est appliqué dans le notebook est néanmoins discuté en section 3.10, car il a une incidence sur l'interprétation des performances.

La capture suivante montre la répartition des huit classes après l'application de SMOTE.

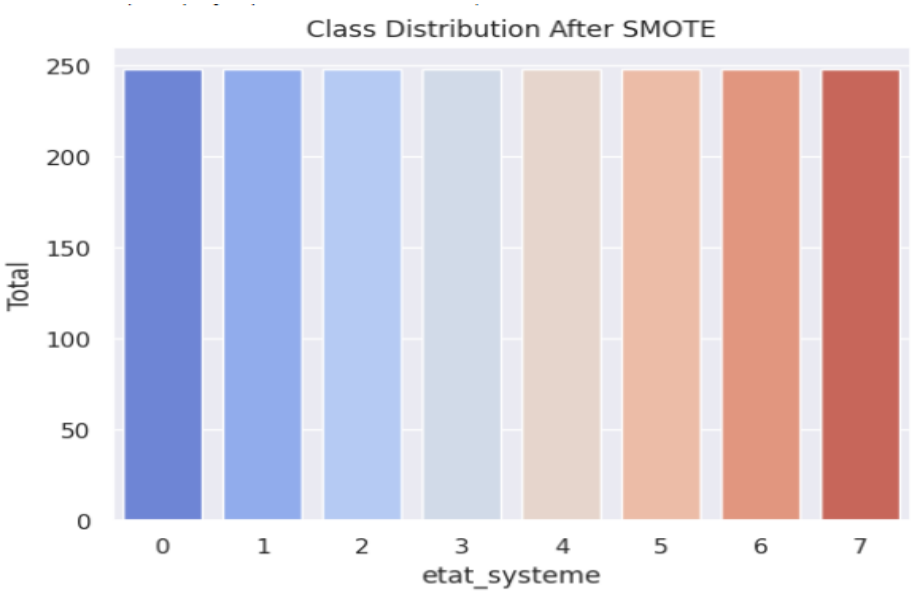

Figure 3.6–Répartition des classes après équilibrage par SMOTE.

3.4.3 Standardisation et séparation apprentissage-test

Les quatre variables d'entrée sont standardisées à l'aide de StandardScaler, puis le jeu obtenu est séparé en 80 % pour l'apprentissage et 20 % pour le test, avec stratification et random_state = 42. Le jeu d'apprentissage contient 1 587 observations et le jeu de test 397 observations.

Tableau 3-6– Dimensions des ensembles utilisés pour l'apprentissage et le test.

Ensemble	Nombre d'observations	Variables d'entrée
Apprentissage	1 587	4
Test	397	4

3.5 Modèle de référence : Random Forest

Afin d'évaluer objectivement l'apport de l'optimisation automatique présentée en section 3.6, une première Random Forest est entraînée avec une configuration explicitement fixée. Ce modèle de référence fournit un point de comparaison indispensable pour juger si la recherche d'hyperparamètres améliore réellement la performance sur un même jeu de test.

Tableau 3-7– Hyperparamètres de la Random Forest de référence.

Hyperparamètre	Valeur
n_estimators	500
max_depth	None
min_samples_split	2
min_samples_leaf	1
max_features	sqrt
criterion	gini
bootstrap	False
class_weight	None
random_state	42

La capture suivante présente les résultats obtenus par la Random Forest de référence sur le jeu de test.

```
Accuracy : 0.9597
Macro F1 : 0.9597

Classification Report:
                                          precision    recall  f1-score   support

                     0 - FUITE_IMPORTANTE       1.00      0.98      0.99        49
                             1 - NORMAL         1.00      0.92      0.96        49
                        2 - FUITE_FAIBLE        0.91      1.00      0.95        50
    3 - FUITE_FAIBLE + RESERVOIR_PRESQUE_VIDE   0.91      0.98      0.94        50
4 - FUITE_IMPORTANTE + RESERVOIR_PRESQUE_VIDE   1.00      1.00      1.00        50
          5 - NORMAL + RESERVOIR_PRESQUE_VIDE   0.98      0.90      0.94        49
                 6 - NORMAL + RESERVOIR_VIDE    0.96      0.94      0.95        50
           7 - FUITE_FAIBLE + RESERVOIR_VIDE    0.94      0.96      0.95        50

                                    accuracy                        0.96       397
                                   macro avg    0.96      0.96      0.96       397
                                weighted avg    0.96      0.96      0.96       397
```

Figure 3.7–Capture – Résultats de la Random Forest de référence sur le jeu de test.

L'analyse par classe montre déjà une séparation excellente pour plusieurs états : la classe FUITE_IMPORTANTE + RESERVOIR_PRESQUE_VIDE est reconnue sans aucune erreur sur les 50 observations de test correspondantes. Les erreurs résiduelles se concentrent principalement entre les états NORMAL et FUITE_FAIBLE, ainsi qu'entre les états partageant une même condition de réservoir (presque vide ou vide).

3.6 Optimisation des modèles avec Optuna

3.6.1 Protocole d'optimisation

Optuna est utilisé pour rechercher automatiquement les meilleurs hyperparamètres de trois modèles : Random Forest, Gradient Boosting et XGBoost. La recherche comporte 30 essais.

Chaque essai est évalué par validation croisée stratifiée à cinq plis. Le F1-score macro moyen est retenu comme critère d'évaluation afin de considérer les huit classes de manière équilibrée. La méthode TPE guide progressivement la recherche vers les configurations donnant les meilleurs résultats.

3.6.2 Configuration retenue par Optuna

Le meilleur essai enregistré est l'essai n° 12. Il sélectionne une Random Forest et obtient un F1-score macro moyen de validation croisée égal à 0,960979.

Tableau 3-8– Hyperparamètres du meilleur essai Optuna.

Hyperparamètre	Valeur retenue
Classifieur	Random Forest
n_estimators	981
max_depth	30
min_samples_split	2
min_samples_leaf	1
max_features	sqrt
criterion	entropy
bootstrap	True
class_weight	None

Les captures suivantes présentent le meilleur essai obtenu avec Optuna et les informations associées à la configuration retenue.

	Trial	CV Macro F1-score	Classifier
12	12	0.9610	RandomForest
10	10	0.9591	RandomForest
11	11	0.9585	RandomForest
21	21	0.9585	RandomForest
22	22	0.9585	RandomForest

Figure 3.8–Capture – Classement des meilleurs essais obtenus avec Optuna.

```
Best classifier:
RandomForest

Best CV Macro F1:
0.961

Best parameters:
{'classifier': 'RandomForest', 'rf_n_estimators': 981, 'rf_max_depth': 30, 'rf_min_samples_split': 2, 'rf_min_samples_leaf': 1, 'rf_max_features': 'sqrt', 'rf_criterion': 'entropy', 'rf_bootstrap': True, 'rf_class_weight': None}
```

Figure 3.9–Capture – Configuration du meilleur modèle sélectionné par Optuna.

3.7 Évaluation et interprétation des résultats

3.7.1 Comparaison globale des deux configurations

Le modèle retenu par Optuna est ensuite réentraîné sur l'ensemble d'apprentissage et évalué sur le même jeu de test que le modèle de référence.

Les deux configurations sont comparées sur 397 observations à l'aide de quatre métriques : exactitude, précision macro, rappel macro et F1-score macro. La matrice de confusion et les résultats par classe complètent cette évaluation en permettant d'identifier les principales erreurs de classification.

Tableau 3-9 – Comparaison des performances sur le jeu de test.

Modèle	Accuracy	Précision macro	Rappel macro	F1 macro
Random Forest de référence	0,9597	0,9618	0,9595	0,9597
Random Forest issue d'Optuna	0,9572	0,9593	0,9570	0,9572

Les deux configurations obtiennent des performances très proches et globalement excellentes.

3.7.2 Choix final

La capture suivante confirme le modèle retenu à l'issue de la procédure d'optimisation.

```
Final model      : Optimized Random Forest - Optuna
Accuracy         : 0.9572
Macro Precision  : 0.9593
Macro Recall     : 0.957
Macro F1-score   : 0.9572
```

Figure 3.10— Résultat du modèle final retenu après optimisation

Toutefois, le modèle issu d'Optuna est retenu comme modèle final, car sa sélection repose sur une validation croisée stratifiée à cinq plis. Contrairement à une évaluation basée sur une seule séparation des données, cette procédure évalue successivement chaque configuration sur plusieurs sous-ensembles de validation issus de l'ensemble d'apprentissage. Elle permet ainsi de sélectionner une configuration présentant un comportement plus régulier sur différentes partitions des données.

La configuration retenue par Optuna a obtenu un F1-score macro moyen de 0,960979 en validation croisée. Elle est ensuite réentraînée sur l'ensemble d'apprentissage puis évaluée sur le jeu de test final. L'écart observé avec le modèle de référence reste faible ; le choix du modèle Optuna repose donc principalement sur la méthode de sélection plus systématique utilisée lors de l'optimisation, et non sur une amélioration du score de test.\

3.7.3 Analyse par classe et matrice de confusion

L'analyse détaillée du modèle issu d'Optuna montre que la majorité des observations sont classées sur la diagonale de la matrice de confusion. La classe FUITE_IMPORTANTE + RESERVOIR_PRESQUE_VIDE est correctement reconnue pour les 50 observations du jeu de test. Les principales erreurs apparaissent entre des états proches, notamment NORMAL et FUITE_FAIBLE, ainsi qu'entre les classes partageant la même condition de réservoir.
La figure 3.11 présente la matrice de confusion du modèle Random Forest issu d'Optuna.

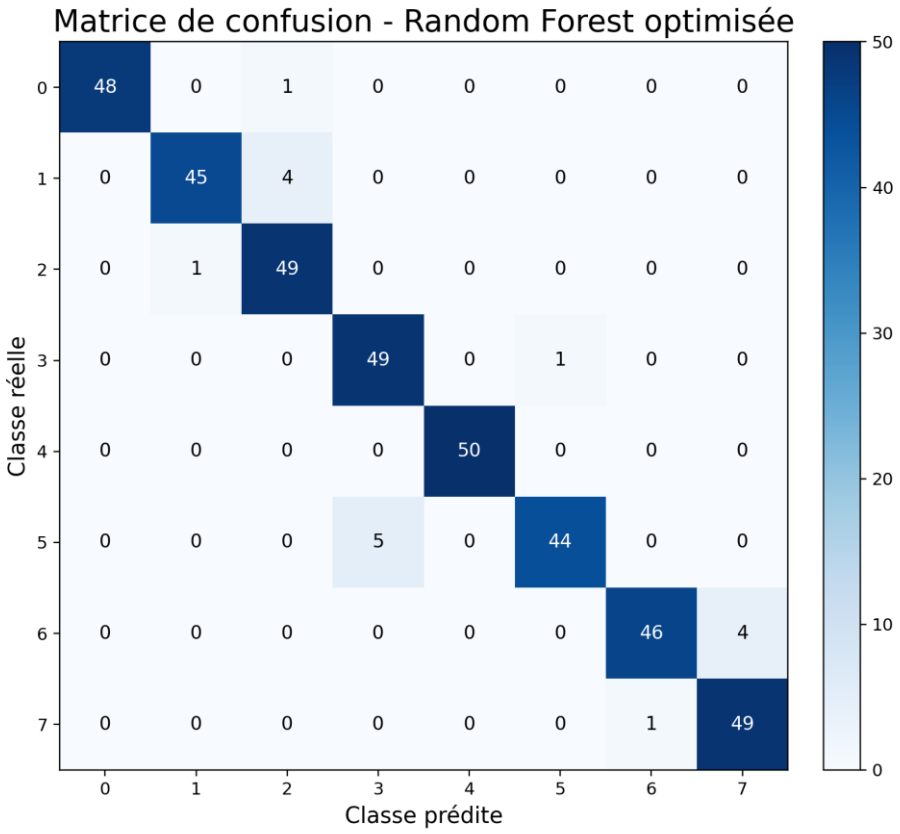

Figure 3.11– Matrice de confusion de la Random Forest issue d'Optuna.

Tableau 3-10— Rapport de classification du modèle Random Forest issu d'Optuna.

Classe	Précision	Rappel	F1-score	Support
FUITE_IMPORTANTE	1,00	0,98	0,99	49
NORMAL	0,98	0,92	0,95	49
FUITE_FAIBLE	0,91	0,98	0,94	50
FUITE_FAIBLE + RESERVOIR_PRESQUE_VIDE	0,91	0,98	0,94	50
FUITE_IMPORTANTE + RESERVOIR_PRESQUE_VIDE	1,00	1,00	1,00	50
NORMAL + RESERVOIR_PRESQUE_VIDE	0,98	0,90	0,94	49
NORMAL + RESERVOIR_VIDE	0,98	0,92	0,95	50
FUITE_FAIBLE + RESERVOIR_VIDE	0,92	0,98	0,95	50

Les métriques calculées pour chaque classe complètent l'accuracy globale en montrant quels états sont les plus difficiles à distinguer. Cette analyse est particulièrement utile dans un problème multiclasse où les classes ne sont pas représentées de manière équilibrée.

3.8 Sauvegarde et intégration du modèle

Le modèle retenu dans le notebook est sauvegardé avec joblib afin d'être réutilisé par l'application. Le package final regroupe le classifieur, le StandardScaler, l'ordre des quatre variables d'entrée, les noms des classes ainsi que le codage de l'état de la pompe. Cette organisation garantit que les nouvelles observations sont traitées avec le même prétraitement que celui utilisé pendant l'apprentissage.

La capture suivante confirme la sauvegarde du modèle et de ses éléments de prétraitement.

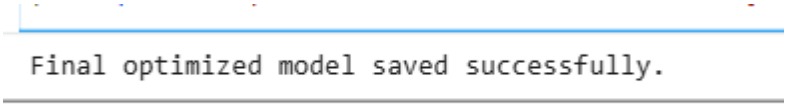

Figure 3.12– content/smartleak_ml_package.joblib

Après sauvegarde, le modèle est rechargé et réévalué sur le jeu de test. L'accuracy et le F1-score macro restent égaux à 0,9572, ce qui confirme que la sérialisation n'a pas modifié les prédictions. Le modèle est ensuite intégré au back-end Flask de l'application de supervision afin de réaliser l'inférence à partir des données reçues.
Après rechargement, les performances du modèle sont vérifiées sur le même jeu de test.

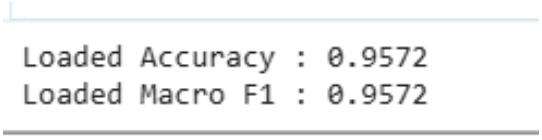

Figure 3.13– Vérification des performances après rechargement du modèle.

3.9 Validation fonctionnelle du système développé

3.9.1 Principe de validation

La validation réalisée à ce stade porte sur l'intégration logicielle. Elle ne correspond pas à une nouvelle campagne d'acquisition : des valeurs reprises de la même base expérimentale sont réinjectées depuis un script Python exécuté sous VS Code. Le serveur Flask reçoit les données, applique le prétraitement associé au modèle, exécute la classification puis met le résultat à disposition du tableau de bord.

La vérification porte ainsi sur trois éléments distincts : le bon traitement de la requête par le serveur, la classe produite par le modèle et la cohérence de l'état affiché sur l'interface. Les captures montrent des requêtes POST vers /api/esp32/data et des requêtes GET vers /api/data utilisées pour l'actualisation du tableau de bord.

Tableau 3-11— Scénarios concordants utilisés pour la validation fonctionnelle.

Scénario	Débit	Pression	Niveau	Classe prédite	Concordance
NORMAL	6,5 L/min	0,300 bar	80 %	NORMAL	OUI
FUITE_FAIBLE	3,2 L/min	0,275 bar	70 %	FUITE_FAIBLE	OUI
FUITE_IMPORTANTE	5,8 L/min	0,245 bar	65 %	FUITE_IMPORTANTE	OUI
NORMAL + RESERVOIR_VIDE	0,0 L/min	0,000 bar	2 %	NORMAL + RESERVOIR_VIDE	OUI
FUITE_FAIBLE + RESERVOIR_VIDE	1,5 L/min	0,270 bar	2 %	FUITE_FAIBLE + RESERVOIR_VIDE	OUI

3.9.2 Vérification de la communication et de l'affichage

Les journaux du serveur et le terminal de test confirment le traitement des requêtes avec un code HTTP 200 pour les scénarios présentés. Les réponses du modèle sont ensuite reprises par l'interface, ce qui permet de vérifier la continuité du flux de données entre le client de test, Flask, le classifieur et le tableau de bord.

La figure 3.14 présente les échanges observés entre le client de test, le serveur Flask et le modèle de classification.

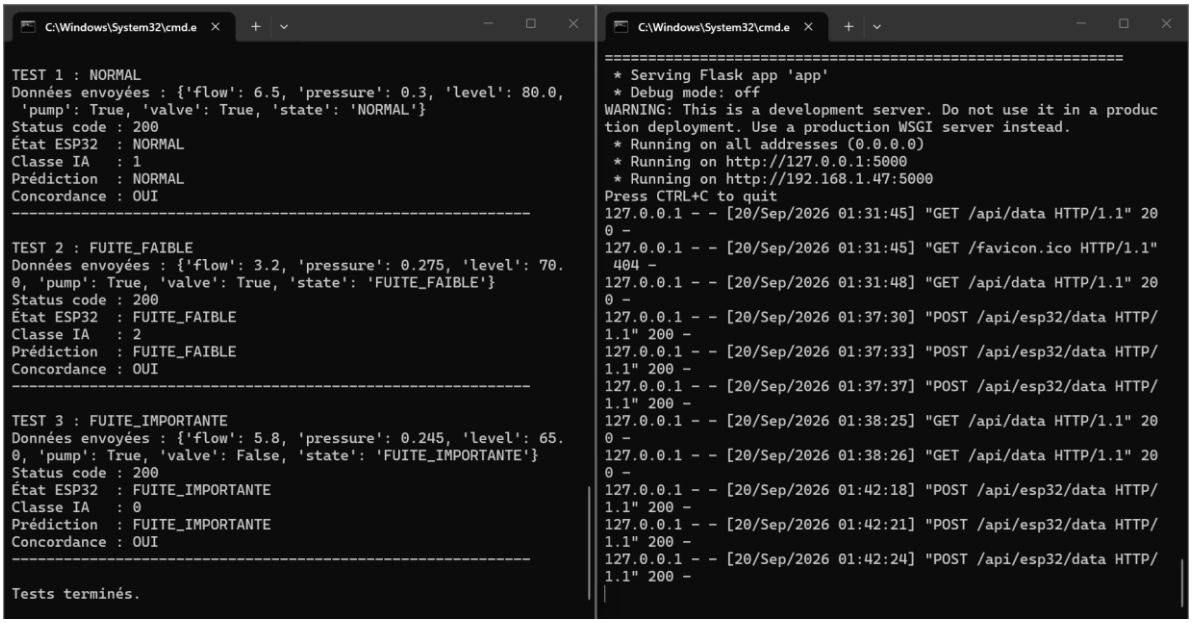

Figure 3.14– Validation de la communication entre le client de test, le serveur Flask et le modèle de classification.

Les trois états principaux sont correctement restitués sur l'interface. Les figures 3.15 à 3.17 montrent respectivement le fonctionnement normal, la fuite faible et la fuite importante, avec concordance entre les valeurs envoyées, la classe prédite et l'état affiché.

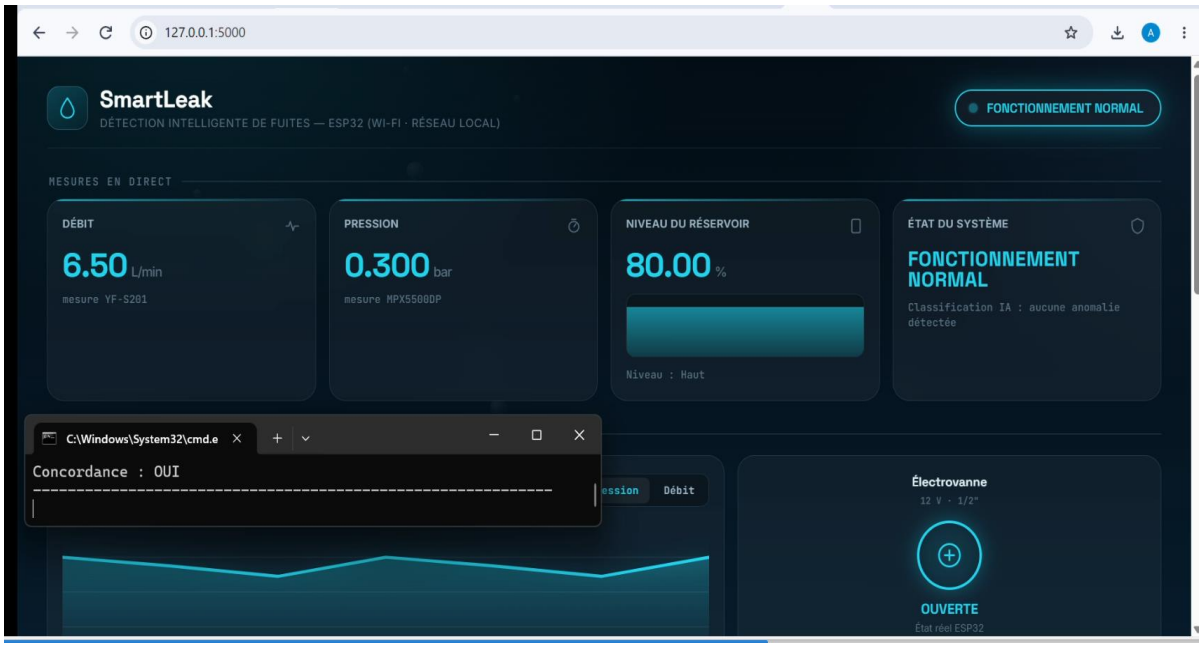

Figure 3.15— Validation du scénario de fonctionnement normal sur le dashboard.

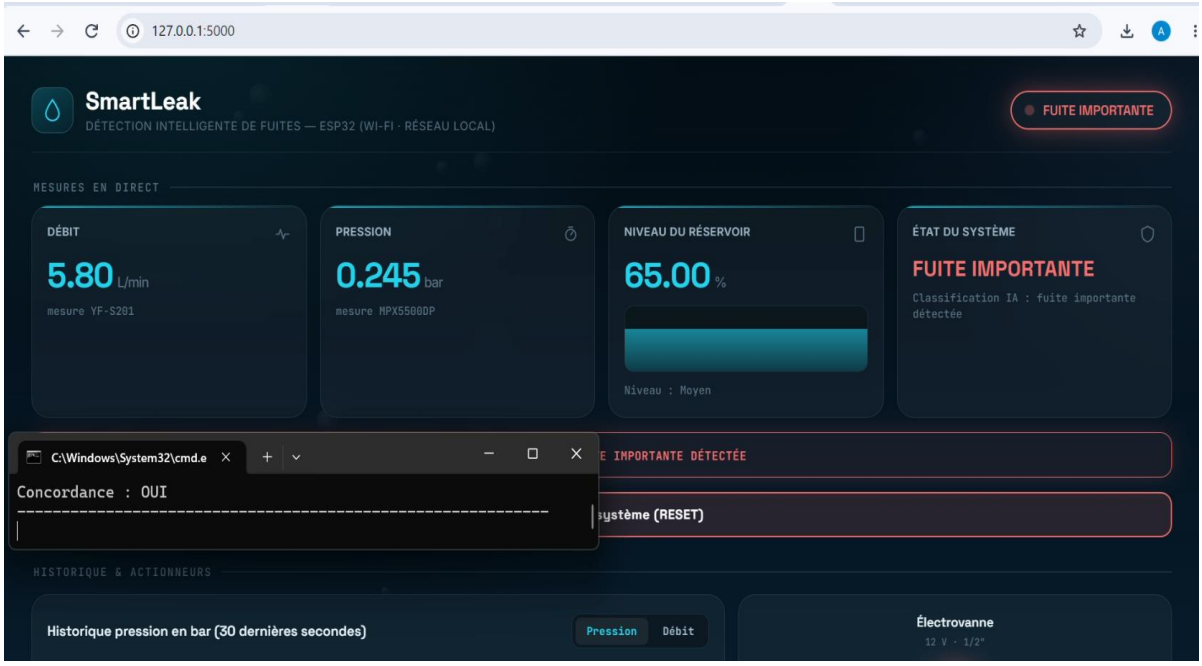

Figure 3.16— Validation du scénario de fuite importante sur l'interface de supervision



Figure 3.17– Validation du scénario de fuite faible sur l'interface de supervision

3.9.3 Vérification des états associés au réservoir vide

Deux scénarios complémentaires sont utilisés pour vérifier les classes associées à un niveau critique du réservoir. Le terminal de test montre une prédiction conforme pour NORMAL + RESERVOIR_VIDE et pour FUITE_FAIBLE + RESERVOIR_VIDE. La figure 3.19 illustre également l'affichage du second cas sur l'interface de supervision.

La figure 3.18 présente les résultats des deux scénarios associés à un réservoir vide.

```
TEST 4 : NORMAL + RESERVOIR_VIDE
Données envoyées : {'flow': 0.0, 'pressure': 0.0, 'level': 2.0, 'pump': False, 'valve': True, 'state': 'NORMAL'}
Status code : 200
État envoyé : NORMAL + RESERVOIR_VIDE
État ESP32  : NORMAL
Classe IA   : 6
Prédiction  : NORMAL + RESERVOIR_VIDE
Concordance : OUI
----------------------------------------------------------------

TEST 5 : FUITE_FAIBLE + RESERVOIR_VIDE
Données envoyées : {'flow': 1.5, 'pressure': 0.27, 'level': 2.0, 'pump': False, 'valve': False, 'state': 'FUITE_FAIBLE'}
Status code : 200
État envoyé : FUITE_FAIBLE + RESERVOIR_VIDE
État ESP32  : FUITE_FAIBLE
Classe IA   : 7
Prédiction  : FUITE_FAIBLE + RESERVOIR_VIDE
Concordance : OUI
----------------------------------------------------------------
```

Figure 3.18–Validation des scénarios NORMAL + RESERVOIR_VIDE et FUITE_FAIBLE + RESERVOIR_VIDE.

La figure 3.19 présente l'affichage du scénario FUITE_FAIBLE + RESERVOIR_VIDE sur l'interface de supervision.

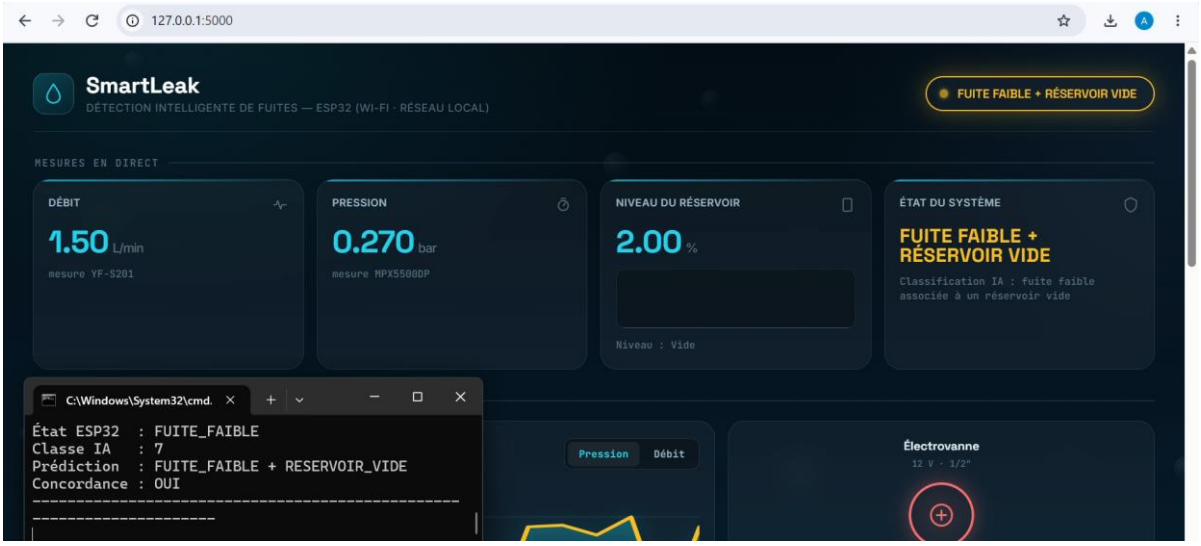

Figure 3.19– Affichage du scénario FUITE_FAIBLE + RESERVOIR_VIDE sur le dashboard.

3.9.4 Portée de la validation fonctionnelle

Ces essais confirment le fonctionnement de la chaîne logicielle locale : transmission des valeurs, exécution du modèle, retour de la classe et mise à jour de l'interface. Ils démontrent donc l'intégration effective du modèle dans l'application de supervision, au-delà de son évaluation dans le notebook.

Cette validation doit toutefois être distinguée d'une acquisition physique continue provenant directement de l'ESP32. Dans les captures présentées, les scénarios sont envoyés

par le client de test sous VS Code ; la validation concerne donc principalement le fonctionnement du modèle et de l'application dans l'environnement local utilisé.

3.10 Discussion critique et limites

3.10.1 Limite liée à l'ordre du prétraitement

Le principal point méthodologique concerne l'ordre des opérations exécutées dans le notebook. SMOTE est appliqué avant la séparation apprentissage-test et le StandardScaler est ajusté avant ce découpage. Des informations provenant de l'ensemble complet interviennent ainsi dans la génération des observations synthétiques et dans le calcul des paramètres de standardisation avant la constitution du jeu de test.

Cette organisation introduit un risque de fuite d'information et peut conduire à une estimation trop favorable des performances de généralisation. Les scores présentés dans ce chapitre sont conservés parce qu'ils correspondent au protocole réellement exécuté. Pour une évaluation plus stricte, la séparation apprentissage-test devrait être effectuée en premier, puis SMOTE et la standardisation appliquée uniquement à l'ensemble d'apprentissage, idéalement à l'intérieur d'un pipeline de validation croisée.

3.10.2 Exemple de non-concordance

La validation complémentaire met également en évidence un cas où la chaîne de communication fonctionne correctement mais où la décision du modèle ne correspond pas à l'état attendu. Pour le scénario NORMAL + RESERVOIR_PRESQUE_VIDE, la requête est acceptée par le serveur (HTTP 200), mais le modèle prédit FUITE_FAIBLE + RESERVOIR_PRESQUE_VIDE ; la concordance est donc indiquée « NON ».

La figure 3.20 présente le cas de non-concordance observé lors de la validation complémentaire.

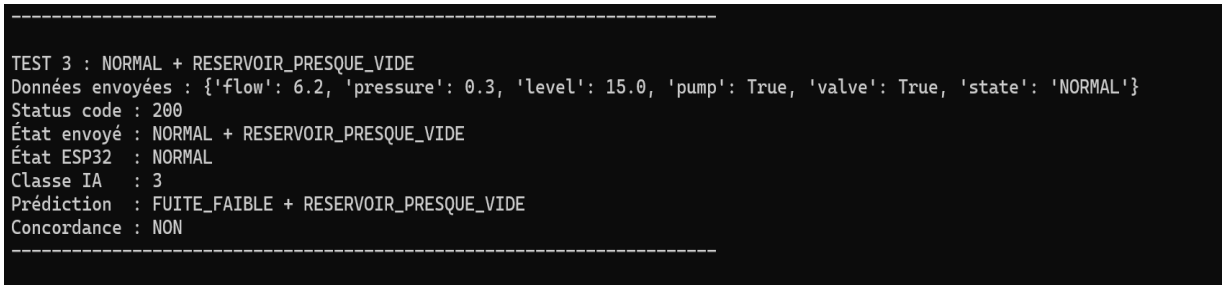

Figure 3.20— Exemple de non-concordance entre l'état attendu et la classe prédite.

Ce résultat ne traduit pas une erreur de transmission ou d'affichage : il correspond à une erreur de classification. Il met en évidence une frontière de décision encore imparfaite entre deux états proches et confirme l'intérêt d'analyser les erreurs, et non uniquement les cas correctement classés.

3.10.3 Représentativité et généralisation

La base a été constituée sur une maquette expérimentale et certaines classes étaient initialement peu représentées, notamment celles associées au réservoir vide. Les performances obtenues caractérisent donc les scénarios couverts par cette base et par le protocole appliqué. Elles ne permettent pas, à ce stade, d'affirmer un comportement identique sur un réseau réel soumis à d'autres conditions hydrauliques, à du bruit de mesure, à des variations de consommation ou à des scénarios absents de l'apprentissage.

3.10.4 Apport du travail réalisé

L'intérêt principal du travail réside dans la continuité de la démarche : les données issues du prototype sont préparées et analysées, plusieurs configurations de classification sont évaluées, le modèle est sauvegardé avec son prétraitement puis intégré dans une application de supervision. Cette chaîne complète permet de relier l'expérimentation, le machine learning et l'IoT dans un même système de validation, tout en identifiant explicitement les limites qui devront être corrigées lors d'une future évaluation indépendante.

Perspectives

Une première évolution consiste à intégrer directement la dimension temporelle dans le modèle d'apprentissage. Dans le système actuel, la persistance d'un comportement anormal est vérifiée par une logique programmée sur une fenêtre de 10 s. À terme, cette temporalité pourrait être exploitée par le modèle lui-même à partir de séquences successives de mesures.

Une seconde perspective concerne l'analyse du comportement de consommation à plus long terme. L'enregistrement continu et horodaté des données sur plusieurs jours ou plusieurs semaines permettrait d'établir un profil de fonctionnement habituel. Le système pourrait alors comparer un comportement observé à cet historique afin de mieux distinguer une consommation inhabituelle mais légitime d'une fuite réelle.

Enfin, l'exploitation de séries temporelles plus longues permettrait d'introduire une dimension prédictive dans le système. L'objectif ne serait plus seulement de classifier l'état courant, mais

également d'anticiper certaines dérives, d'estimer l'évolution future des grandeurs hydrauliques et d'identifier plus précocement des comportements susceptibles d'annoncer une fuite. Cette évolution nécessiterait une base de données plus riche, horodatée et représentative, ainsi que l'étude de modèles adaptés aux données séquentielles.

Conclusion

Ce chapitre a permis de mettre en évidence la capacité du modèle retenu à distinguer les différents états de fonctionnement à partir des données expérimentales. Les résultats obtenus montrent des performances globalement élevées, tout en révélant certaines confusions entre des classes proches.

L'intégration du modèle dans l'application de supervision a également permis de vérifier son fonctionnement dans la chaîne de prédiction et d'affichage. Les limites identifiées, notamment celles liées au protocole de prétraitement et à certains cas de non-concordance, soulignent toutefois la nécessité d'une validation plus rigoureuse sur des données indépendantes.

Conclusion générale

Ce travail a permis d'étudier l'apport de l'apprentissage automatique pour la reconnaissance de différents états de fonctionnement d'un prototype expérimental de détection des fuites d'eau.

À partir des données acquises, plusieurs étapes de traitement et de classification ont été mises en œuvre. Les modèles obtenus présentent des performances élevées sur le jeu de données utilisé, avec un F1-score macro de 0,9597 pour la Random Forest de référence et de 0,9572 pour le modèle retenu après optimisation avec Optuna.

L'analyse détaillée a toutefois montré certaines confusions entre des états présentant des caractéristiques proches.

L'intégration du modèle dans une plateforme locale de supervision a ensuite permis de vérifier le fonctionnement de la chaîne de prédiction et d'affichage des résultats.

Ces résultats restent limités aux conditions expérimentales et aux scénarios représentés dans la base de données. Une validation sur des données indépendantes et dans des conditions plus variées sera nécessaire avant toute généralisation à des installations hydrauliques réelles.

Ce travail constitue ainsi une première base expérimentale pour le développement d'un système de surveillance intelligent, dont la robustesse pourra être améliorée par l'enrichissement des données et l'élargissement des conditions de validation.

Références bibliographiques

[1] M. T. Mustapha, I. Ozsahin, and D. U. Ozsahin, “Introduction to machine learning and artificial intelligence,” in Artificial Intelligence and Image Processing in Medical Imaging, W. A. Zgallai and D. U. Ozsahin, Eds. Academic Press, 2024, pp. 1–19, doi: 10.1016/B978-0-323-95462-4.00001-7.

[2] O. Colliot, “A Non-technical Introduction to Machine Learning,” in Machine Learning for Brain Disorders, O. Colliot, Ed. New York, NY: Humana, 2023, ch. 1, pp. 3–23, doi: 10.1007/978-1-0716-3195-9_1.

[3] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[4] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions,” SN Computer Science, vol. 2, no. 3, art. 160, 2021, doi: 10.1007/s42979-021-00592-x.

[5] B. Bala and S. Behal, “A Brief Survey of Data Preprocessing in Machine Learning and Deep Learning Techniques,” in 2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2024, pp. 1755–1762, doi: 10.1109/I-SMAC61858.2024.10714767.

[6] A. Apicella, F. Isgrò, and R. Prevete, “Don’t push the button! Exploring data leakage risks in machine learning and transfer learning,” Artificial Intelligence Review, vol. 58, art. 339, 2025, doi: 10.1007/s10462-025-11326-3.

[7] N. C. Spies and D. P. Ng, “Performance metrics for machine learning solutions in laboratory medicine,” Laboratory Medicine, vol. 56, no. 6, pp. 597–607, 2025, doi: 10.1093/labmed/lmaf013.

[8] S. Arlot and A. Celisse, “A survey of cross-validation procedures for model selection,” Statistics Surveys, vol. 4, pp. 40–79, 2010, doi: 10.1214/09-SS054.

[9] L. Breiman, “Random Forests,” Machine Learning, vol. 45, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[10] C. Cortes and V. Vapnik, “Support-Vector Networks,” Machine Learning, vol. 20, pp. 273–297, 1995, doi: 10.1007/BF00994018.

[11] I. Niño-Adan, I. Landa-Torres, E. Portillo, and D. Manjarres, “Influence of statistical feature normalisation methods on K-Nearest Neighbours and K-Means in the context of industry 4.0,” Engineering Applications of Artificial Intelligence, vol. 111, art. 104807, 2022, doi: 10.1016/j.engappai.2022.104807.

[12] Z. Wu and X. Zhang, “Feature Rescaling of Support Vector Machines,” Tsinghua Science and Technology, vol. 16, no. 4, pp. 414–421, 2011, doi: 10.1016/S1007-0214(11)70060-8.

[13] C. Miller, T. Portlock, D. M. Nyaga, and J. M. O’Sullivan, “A review of model evaluation metrics for machine learning in genetics and genomics,” Frontiers in Bioinformatics, vol. 4, art. 1457619, 2024, doi: 10.3389/fbinf.2024.1457619.

[14] O. A. Montesinos López, A. Montesinos López, and J. Crossa, “Overfitting, Model Tuning, and Evaluation of Prediction Performance,” in Multivariate Statistical Machine Learning Methods for Genomic Prediction. Cham: Springer, 2022, ch. 4, pp. 109–139, doi: 10.1007/978-3-030-89010-0_4.

[15] X. Guan and H. Burton, “Bias-variance tradeoff in machine learning: Theoretical formulation and implications to structural engineering applications,” Structures, vol. 46, pp. 17–30, 2022, doi: 10.1016/j.istruc.2022.10.004.

[16] D. Zaman, M. K. Tiwari, A. K. Gupta, and D. Sen, “A review of leakage detection strategies for pressurised pipeline in steady-state,” Engineering Failure Analysis, vol. 109, art. 104264, 2020, doi: 10.1016/j.engfailanal.2019.104264.

[17] L. Romero-Ben, D. Alves, J. Blesa, G. Cembrano, V. Puig, and E. Duviella, “Leak detection and localization in water distribution networks: Review and perspective,” Annual Reviews in Control, vol. 55, pp. 392–419, 2023, doi: 10.1016/j.arcontrol.2023.03.012.

[18] Y. Wu, S. Liu, and Z. Kapelan, “Addressing data limitations in leakage detection of water distribution systems: Data creation, data requirement reduction, and knowledge transfer,” Water Research, vol. 267, art. 122471, 2024, doi: 10.1016/j.watres.2024.122471.

[19] C. Sun, B. Parellada, V. Puig, and G. Cembrano, “Leak Localization in Water Distribution Networks Using Pressure and Data-Driven Classifier Approach,” Water, vol. 12, no. 1, art. 54, 2020, doi: 10.3390/w12010054.

[20] M. Shekofteh, M. Jalili Ghazizadeh, and J. Yazdi, “A methodology for leak detection in water distribution networks using graph theory and artificial neural network,” Urban Water Journal, vol. 17, no. 6, pp. 525–533, 2020, doi: 10.1080/1573062X.2020.1797832.

[21] F. Taghlabi, L. Sour, and A. Agoumi, “Prelocalization and leak detection in drinking water distribution networks using modeling-based algorithms: a case study for the city of Casablanca (Morocco),” Drinking Water Engineering and Science, vol. 13, pp. 29–41, 2020, doi: 10.5194/dwes-13-29-2020.

[22] L. Romero, J. Blesa, V. Puig, G. Cembrano, and C. Trapiello, “First Results in Leak Localization in Water Distribution Networks using Graph-Based Clustering and Deep Learning,” IFAC-PapersOnLine, vol. 53, no. 2, pp. 16691–16696, 2020, doi: 10.1016/j.ifacol.2020.12.1104.

[23] D. Levinas, G. Perelman, and A. Ostfeld, “Water Leak Localization Using High-Resolution Pressure Sensors,” Water, vol. 13, no. 5, art. 591, 2021, doi: 10.3390/w13050591.

[24] N. Mashhadi, I. Shahrour, N. Attoue, J. El Khattabi, and A. Aljer, “Use of Machine Learning for Leak Detection and Localization in Water Distribution Systems,” Smart Cities, vol. 4, no. 4, pp. 1293–1315, 2021, doi: 10.3390/smartcities4040069.

[25] I. Lučin, B. Lučin, Z. Čarija, and A. Sikirica, “Data-Driven Leak Localization in Urban Water Distribution Networks Using Big Data for Random Forest Classifier,” Mathematics, vol. 9, no. 6, art. 672, 2021, doi: 10.3390/math9060672.

[26] R. Vanijjirattikhan et al., “AI-based acoustic leak detection in water distribution systems,” Results in Engineering, vol. 15, art. 100557, 2022, doi: 10.1016/j.rineng.2022.100557.

[27] I. A. Tijani, S. Abdelmageed, A. Fares, K. H. Fan, Z. Y. Hu, and T. Zayed, “Improving the leak detection efficiency in water distribution networks using noise loggers,” Science of the Total Environment, vol. 821, art. 153530, 2022, doi: 10.1016/j.scitotenv.2022.153530.

[28] S. G. Vrachimis, M. S. Kyriakou, D. G. Eliades, and M. M. Polycarpou, “LeakDB: A benchmark dataset for leakage diagnosis in water distribution networks,” in Proc. WDSA/CCWI Joint Conference, 2018.

[29] S. G. Vrachimis et al., “Battle of the Leakage Detection and Isolation Methods,” Journal of Water Resources Planning and Management, vol. 148, no. 12, art. 04022068, 2022, doi: 10.1061/(ASCE)WR.1943-5452.0001601.

[30] H. M. Tornyeviadzi, H. Mohammed, and R. Seidu, “Semi-supervised anomaly detection methods for leakage identification in water distribution networks: A comparative study,” Machine Learning with Applications, vol. 14, art. 100501, 2023, doi: 10.1016/j.mlwa.2023.100501.

[31] D. P. Sousa, R. Du, J. M. Barros da Silva Jr., C. C. Cavalcante, and C. Fischione, “Leakage detection in water distribution networks using machine-learning strategies,” Water Supply, vol. 23, no. 3, pp. 1115–1126, 2023, doi: 10.2166/ws.2023.054.

[32] M. M. Rajabi, P. Komeilian, X. Wan, and R. Farmani, “Leak detection and localization in water distribution networks using conditional deep convolutional generative adversarial networks,” Water Research, vol. 238, art. 120012, 2023, doi: 10.1016/j.watres.2023.120012.

[33] N. T. Mücke, P. Pandey, S. Jain, S. M. Bohté, and C. W. Oosterlee, “A Probabilistic Digital Twin for Leak Localization in Water Distribution Networks Using Generative Deep Learning,” Sensors, vol. 23, no. 13, art. 6179, 2023, doi: 10.3390/s23136179.

[34] Y. Wu et al., “Advancing deep learning-based acoustic leak detection methods towards application for water distribution systems from a data-centric perspective,” Water Research, vol. 261, art. 121999, 2024, doi: 10.1016/j.watres.2024.121999.

[35] J. Luo, C. Wang, J. Yang, and X. Zhong, “A Transformer-Based Approach to Leakage Detection in Water Distribution Networks,” Sensors, vol. 24, no. 19, art. 6294, 2024, doi: 10.3390/s24196294.

[36] S. Jun, D. Jung, and K. Lansey, “Three-dimensional convolutional neural network for leak detection and localization in smart water distribution systems,” Water Research X, vol. 25, art. 100264, 2024, doi: 10.1016/j.wroa.2024.100264.

[37] X. Li and Y. Wu, “A Convolutional Graph Neural Network Model for Water Distribution Network Leakage Detection Based on Segment Feature Fusion Strategy,” Water, vol. 16, no. 24, art. 3555, 2024, doi: 10.3390/w16243555.

[38] M. Adraoui et al., “A two-phase approach for leak detection and localization in water distribution systems using wavelet decomposition and machine learning,” Computers & Industrial Engineering, vol. 197, art. 110534, 2024, doi: 10.1016/j.cie.2024.110534.

[39] Z. Li, Y. Tian, and S. Peng, “Detection and localization of pipeline leakage of water distribution networks based on graph convolutional networks,” Computer-Aided Civil and Infrastructure Engineering, vol. 40, no. 31, pp. 6656–6677, 2025, doi: 10.1111/mice.70173.

[40] Z. Zhang and O. Fink, “Algorithm-informed graph neural networks for leakage detection and localization in water distribution networks,” Reliability Engineering & System Safety, vol. 265, art. 111494, 2026, doi: 10.1016/j.ress.2025.111494.

[41] S. Ranganathan, N. Vinothkumar, and E. Muthuramalingam, “Benchmarking Diverse Learning Architectures for Leak Detection and Localization in Water Distribution Systems,” Journal of Water Process Engineering, art. 110114, 2026, doi: 10.1016/j.jwpe.2026.110114.

[42] C. G. N. Gouveia, “Técnicas de aprendizado de máquina aplicadas à predição de vazamentos em ramais de redes de distribuição de água,” Master’s thesis, Universidade de Brasília, Brasília, Brazil, 2022.

[43] Z. Meng, “Leak Localization in Water Distribution Networks,” Master’s thesis, Delft University of Technology, Delft, The Netherlands, 2022.

[44] C. M. Corzo Delgado, “Multiple leaks location in water distribution networks combining physically-based and machine learning approaches using high-performance computing,” M.Sc. thesis, IHE Delft Institute for Water Education, Delft, The Netherlands, 2022.

[45] T. Green, “Leak Detection and Localization in Water Distribution Networks,” M.A.Sc. thesis, University of Waterloo, Waterloo, Canada, 2026.

[46] D. Han, “Leak Detection and Sensor Optimization in Water Distribution Networks Using Deep Learning,” Master’s thesis, Hong Kong University of Science and Technology, Hong Kong, 2026, doi: 10.14711/thesis-hdl172472.

[47] C. M. Corzo, L. Alfonso, G. Corzo, and D. Solomatine, “Locating Multiple Leaks in Water Distribution Networks Combining Physically Based and Data-Driven Models and High-Performance Computing,” Journal of Water Resources Planning and Management, vol. 149, no. 12, 2023, doi: 10.1061/JWRMD5.WRENG-6005.

[48] T. Reece, “Leak detection with artificial intelligence,” WO Patent WO2020033316A1, Feb. 13, 2020.

[49] “Water supply pipe leakage early warning method and system based on artificial intelligence,” China Patent CN113112126A, Jul. 13, 2021; grant CN113112126B, 2024.

[50] X. Yu and X. Fan, “Systems and methods for water distribution network leakage detection and/or localization,” U.S. Patent Application US20240084561A1, Mar. 14, 2024.

[51] “Apparatus for determining leakage using pipeline characteristics and noise pattern, and learning method thereof,” WO Patent WO2025063582A1, Mar. 27, 2025.

Annexes

Annexe X — Représentation UML de la structure des données expérimentales

Cette annexe présente la représentation UML de la structure des données expérimentales utilisées pour l'apprentissage automatique. Chaque enregistrement de la base correspond à une observation caractérisée par trois variables numériques, l'état de la pompe et l'état du système.

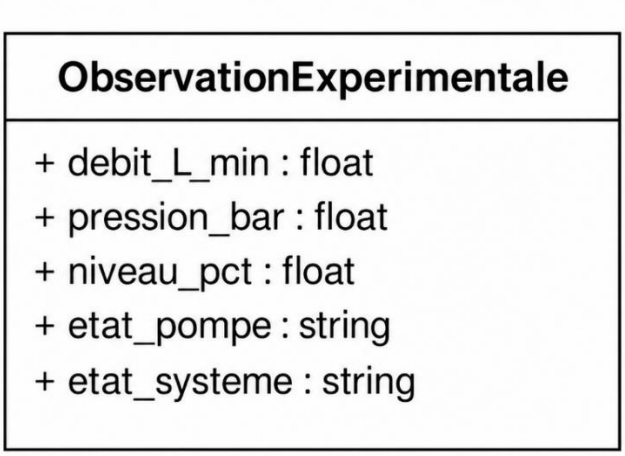

Figure 3.21– – Diagramme de classe UML représentant la structure d'une observation expérimentale.

Les attributs debit_L_min, pression_bar, niveau_pct et etat_pompe constituent les variables d'entrée du modèle de classification. L'attribut etat_systeme correspond à la variable cible. Chaque ligne de la base de données représente une instance de la classe ObservationExperimentale.

Google Colab

L'ensemble des expérimentations liées à l'apprentissage automatique a été réalisé à l'aide de Google Colab, une plateforme cloud permettant d'exécuter du code Python dans un environnement interactif. Elle a été utilisée pour le chargement et la préparation des données, l'entraînement des différents modèles de Machine Learning, ainsi que l'évaluation et la comparaison de leurs performances. Les principales bibliothèques utilisées sont notamment
Pandas, NumPy et Scikit-learn.



Figure 3.22– google Colab

Visual Studio Code

VS Code est un éditeur de code source développé par Microsoft. Il prend en charge plusieurs langages de programmation et intègre des outils de débogage, un terminal ainsi qu'un système d'extensions permettant d'adapter l'environnement aux besoins du projet.

Dans ce travail, VS Code a été utilisé pour développer la partie logicielle de supervision, notamment le serveur local Python/Flask et les fichiers HTML, CSS et JavaScript du tableau de bord. Il a également servi à tester et vérifier les échanges de données entre l'ESP32, le serveur local et l'interface de supervision.



Figure 3.23– Vs Code

ملخص

يركز هذا العمل على تطبيق تقنيات التعلم الآلي في كشف وتصنيف تسربات المياه اعتماداً على بيانات هيدروليكية تجريبية. تم إنشاء قاعدة بيانات انطلاقاً من نموذج هيدروليكي مجهز بأجهزة قياس، بالاعتماد على التدفق والضغط ومستوى الخزان وحالة المضخة. بعد تحضير البيانات وتحليلها، تم اعتماد نموذج الغابة العشوائية Random Forest مع Optuna تحسينه باستخدام على Optuna. ثم دمجه في تطبيق مراقبة يعتمد Flask ، أظهرت النتائج إمكانية تطبيق هذا النهج على النموذج التجريبي.

الكلمات المفتاحية :كشف التسربات، التعلم الآلي، الغابة العشوائية، البيانات الهيدروليكية، المراقبة.

Résumé

Ce travail porte sur l'application de l'apprentissage automatique à la détection et à la classification des fuites d'eau à partir de données hydrauliques expérimentales. Une base de données a été constituée à partir d'une maquette instrumentée en exploitant le débit, la pression, le niveau du réservoir et l'état de la pompe. Après préparation et analyse des données, un modèle de classification de type Random Forest, optimisé avec Optuna, a été retenu puis intégré à une application Flask de supervision. Les résultats obtenus montrent la faisabilité de cette approche sur la maquette expérimentale.

Mots-clés : détection de fuites, apprentissage automatique, Random Forest, données hydrauliques, supervision.

Abstract

This work focuses on the application of machine learning to water leak detection and classification using experimental hydraulic data. A dataset was built from an instrumented hydraulic prototype using flow rate, pressure, reservoir level, and pump status. After data preparation and analysis, a Random Forest classifier optimized with Optuna was selected and integrated into a Flask-based monitoring application. The obtained results demonstrate the feasibility of this approach on the experimental prototype.

Keywords: water leak detection, machine learning, Random Forest, hydraulic data, monitoring.