



REPUBLIQUE DU BURUNDI
MINISTÈRE DE L'ÉDUCATION NATIONALE
ET DE LA RECHERCHE SCIENTIFIQUE



UNIVERSITE DU BURUNDI
INSTITUT DE STATISTIQUE
APPLIQUEE

SYSTEME DE DETECTION DE FRAUDE EN ASSURANCE BASE SUR L'ANALYSE DES DONNEES CLIENTS ET SINISTRES (MACHINE LEARNING)

Réalisé par :

NDAYISHIMIYE Salvator

Matricule : 23/09894

Master II en Actuariat et Finance

EXAMEN DU COURS D'INGENIERIE DES LOGICIELS III

Enseignant responsable : Dr NIBIGIRA Nadine

Année académique : 2024-2025

1 Introduction

La fraude dans le domaine de l'assurance représente un problème sérieux pour les sociétés d'assurance partout dans le monde. Elle désigne tout comportement délibéré visant à obtenir un avantage en trompant, en cachant ou en exagérant un dommage (Viaene, Derrig, Baesens et Dedene 2002). Dans le cas des assurances automobile et santé, cela peut prendre la forme de fausses déclarations d'accidents, de factures surestimées ou d'une complicité entre assurés et réparateurs. Ces pratiques causent des pertes importantes aux assureurs. Ces fraudes sont souvent difficiles à repérer avec les méthodes classiques, et cela entraîne des coûts supplémentaires que les clients honnêtes paient finalement, sous forme d'augmentation des primes d'assurance (Brockett, Xia et Derrig 1998).

Devant ce problème, l'utilisation des techniques d'intelligence artificielle, notamment le machine learning, ouvre de nouvelles possibilités pour repérer et prévenir les fraudes (Phua et al. 2010). En examinant les données des clients et des sinistres, on peut détecter des comportements anormaux ou suspects en comparant les caractéristiques d'un sinistre à des cas connus de fraude ou de déclarations réelles (Jain, Nandakumar et Ross 2016).

1.1 Objectifs de l'étude

1.1.1 Objectif global :

Concevoir et illustrer un système de détection de fraude basé sur le machine learning, en mettant en avant la pertinence de l'analyse des données actuarielles et financières pour améliorer la rentabilité et la viabilité du portefeuille d'assurance.

1.1.2 Objectifs spécifiques :

1. Repérer les informations importantes provenant des données des clients et des incidents qui pourraient signifier un risque de fraude.
2. Expliquer comment les algorithmes d'apprentissage automatique (comme la régression logistique, les arbres de décision, les réseaux de neurones, etc.) peuvent être utilisés sur ces données pour identifier les cas qui suscitent des doutes (Baesens et al., 2003).
3. Comparer l'efficacité d'un modèle de détection de fraude prédictif avec les méthodes traditionnelles.
4. Souligner les bénéfices actuariels et financiers obtenus grâce à la réduction des pertes causées par la fraude.

1.2 Problématique actuarielle et financière

La fraude en assurance est un problème qui se présente sous deux aspects importants :

D'un point de vue actuariel :

1. Elle affecte la manière dont on calcule les primes, car les données utilisées incluent des cas de fraude, ce qui mène à une estimation erronée du risque (Brockett et Xia, 1995).
2. Elle crée une injustice entre les clients, car ceux qui sont honnêtes paient plus cher, subventionnant ainsi indirectement les fraudeurs.
3. Elle perturbe l'équilibre financier du portefeuille et met en danger la capacité de l'assureur à rester solvable à long terme.

D'un point de vue financier :

1. Les pertes liées à la fraude peuvent représenter entre 5 et 10 % des primes payées (Coalition Against Insurance Fraud, 2022).
2. Les coûts liés à la fraude ne se limitent pas aux pertes directes : il faut aussi compter le temps perdu pour enquêter, les frais juridiques et les difficultés dans les relations avec les clients.
3. L'absence de méthodes fiables pour détecter la fraude réduit le bénéfice de l'entreprise et diminue la confiance des clients et des partenaires.

Problème principal : Comment créer un système utilisant l'apprentissage automatique, en utilisant les données des clients et des sinistres, afin de réduire beaucoup les fraudes et ainsi augmenter le profit financier et la précision des portefeuilles d'assurance ?

2 Contexte et justification

2.1 Contexte

Avec l'augmentation de l'utilisation des technologies numériques dans les services financiers et d'assurance, les sociétés d'assurance récoltent aujourd'hui une grande quantité d'informations (appelées big data), provenant de différentes sources :

- des dossiers des clients (âge, métier, historique des sinistres, mode de paiement, etc.),
- des déclarations de sinistres (date, type, montant, circonstances),
- des échanges avec des tiers (réparateurs, hôpitaux, garages agréés, etc.).

Jusqu'à présent, pour détecter les fraudes, on utilisait des règles précises ou des méthodes manuelles (par exemple : vérifier si un client déclare trop de sinistres en peu de temps). Cependant, ces méthodes ont des difficultés à suivre l'évolution des techniques des fraudeurs et à gérer la quantité croissante d'informations (Bolton et Hand, 2002).

Le machine learning devient alors une solution plus efficace :

- Il permet d'apprendre à partir des données passées sur les fraudes et sur les sinistres normaux,
- Il peut repérer des comportements anormaux que l'œil humain pourrait négliger (Ngai et al., 2011),
- Il s'adapte constamment en intégrant de nouveaux cas de fraude.

2.2 Justification

La création d'un système intelligent pour détecter les fraudes est nécessaire pour plusieurs raisons :

1. En termes économiques : cela permet de diminuer les pertes d'argent, d'améliorer le bénéfice de l'entreprise et de protéger ceux qui paient correctement leurs cotisations.
2. En termes actuariels : cela facilite une estimation plus précise des risques, une gestion meilleure des contrats et une meilleure capacité à assurer les engagements.
3. En termes opérationnels : cela permet d'automatiser les vérifications, de gagner du temps pour les gens qui gèrent les accidents et d'augmenter l'efficacité du travail.
4. En termes stratégiques : cela améliore la réputation de l'entreprise, renforce la confiance des clients et des organismes de régulation, et permet de suivre les évolutions technologiques du secteur (Association of Certified Fraud Examiners, 2020).

Ainsi, utiliser le machine learning pour détecter les fraudes n'est pas seulement une idée technologique, mais une nécessité pour assurer la durabilité financière et technique des sociétés d'assurance dans un marché de plus en plus compétitif.

3 Détection de fraude en assurance (machine learning)

3.1 Méthodes traditionnelles et limites

Historiquement, la détection des fraudes reposait sur des règles basées sur l'expertise (par exemple : seuils, règles heuristiques) et sur des contrôles effectués manuellement par des experts en sinistres (par exemple : vérification des documents, enquêtes sur place,

etc.). Ces méthodes sont faciles à mettre en œuvre, mais elles ont des limites importantes face à la complexité et à la grande quantité de données d'aujourd'hui : elles sont rigides, dépendent beaucoup de l'expertise humaine et sont inefficaces pour repérer de nouveaux ou de subtils schémas de fraude (Bolton et Hand, 2002). Les auteurs soulignent également que les méthodes statistiques classiques restent utiles, mais elles ont des difficultés à gérer des volumes importants de données et à détecter des anomalies non linéaires (Bolton et Hand, 2002 ; Viaene et al., 2002).

3.2 Approches statistiques et premières méthodes de machine learning

Les méthodes statistiques comme la régression logistique, l'analyse discriminante ou la PCA ont été les premières utilisées pour aborder ce problème. Elles offrent un moyen facile à comprendre et adaptées lorsque les liens entre les données sont simples et linéaires (Brockett et Xia, 1995 ; Viaene et al., 2002). Toutefois, des études montrent que des outils plus complexes comme les arbres de décision, les forêts aléatoires, les SVM ou les réseaux de neurones donnent souvent de meilleurs résultats pour repérer des motifs difficiles (Baesens et al., 2003 ; Phua et al., 2010). Une analyse approfondie des différentes techniques de classification montre que le choix de la méthode dépend autant du bon traitement des données que de l'algorithme utilisé (Viaene et al., 2002 ; Baesens et al., 2003).

3.3 Méthodes supervisées, non supervisées et semi-supervisées

La littérature reconnaît trois grandes approches :

Supervisé : nécessite des données déjà classées (fraude ou non-fraude). C'est très efficace quand les données sont fiables. Les méthodes utilisées incluent la régression logistique, les arbres de décision, les forêts aléatoires, le boosting en gradient et les réseaux de neurones profonds (Ngai et al. 2011 ; Phua et al. 2010).

Non supervisé (détecteur d'anomalies) : utile quand les cas de fraude sont rares ou difficiles à identifier. Les méthodes employées sont le clustering, l'isolation forest, les méthodes basées sur la densité et les méthodes à voisinage local (Bolton et Hand 2002).

Semi-supervisé / apprentissage actif : combine peu de données étiquetées avec beaucoup de données non étiquetées. C'est pertinent lorsque l'annotation manuelle est coûteuse (Phua et al. 2010).

Phua et al. (2010) et Ngai et al. (2011) ont présenté des synthèses montrant que les approches mixtes, comme la combinaison de règles et de modèles d'apprentissage

automatique, ou l’empilement de modèles, donnent souvent les meilleures performances en pratique.

3.4 Problème du déséquilibre de classes et techniques de correction

La fraude est souvent un problème lié à des classes très déséquilibrées : il y a très peu de cas de fraude par rapport aux cas légitimes. Cela peut influencer négativement l’apprentissage et les indicateurs classiques comme la précision. La littérature propose plusieurs approches pour y faire face :

- Rééquilibrer les données en diminuant le nombre d’exemples de la classe majoritaire ou en créant des exemples synthétiques pour la classe minoritaire (comme avec SMOTE) ;
- Utiliser des algorithmes et des fonctions de perte adaptés, en pondérant les classes ou en assignant des coûts différents ;
- Employer des méthodes d’ensemble qui privilégient la détection des cas rares (comme le bagging ou le boosting avec un échantillonnage ciblé).

Les recherches montrent que traiter correctement ce déséquilibre améliore la détection sans provoquer trop de faux positifs. Cependant, il est essentiel de régler les paramètres avec soin pour éviter de perdre en généralisation.

3.5 Ingénierie des variables (feature engineering) et sources de données

Plusieurs études montrent que la qualité et la diversité des variables sont souvent plus importantes que le choix de l’algorithme. Parmi les variables utiles et efficaces, on trouve l’historique des incidents, la fréquence des déclarations, les montants impliqués, les occurrences simultanées d’acteurs comme les garages ou les médecins, la localisation géographique des événements, la temporalité (saisonnalité, groupes de dates proches), les métadonnées des réclamations (textes libres, photos), ainsi que des variables liées au comportement (retards de paiement, changements d’utilisation) (Ngai et al., 2011 ; Baesens et al., 2003). L’utilisation de données externes comme les bases officielles, les listes interdites, les données téléphoniques ou les données du véhicule peut améliorer les résultats, à condition qu’elle soit autorisée par la loi.

3.6 Évaluation des modèles : métriques pertinentes

Dans un contexte de fraude, la précision seule peut être trompeuse ; la littérature suggère d'utiliser des métriques plus fiables comme la précision, le rappel (sensibilité), le F1-score, la courbe AUC-ROC, la courbe précision-rappel et le coût attendu (coût des faux positifs versus ceux des faux négatifs). L'évaluation doit inclure une analyse du rapport coût-bénéfice pour l'assureur (perte moyenne évitée versus coût des vérifications des alertes). La recommandation principale est de transformer la performance statistique en impact financier concret (Phua et al., 2010 ; Ngai et al., 2011).

3.7 Explicabilité et acceptabilité opérationnelle

Les modèles complexes comme le deep learning ou les ensembles peuvent être difficiles à comprendre ; cependant, pour des décisions importantes comme bloquer un paiement ou lancer une enquête, il est essentiel de savoir pourquoi une décision a été prise. Cela permet aux gestionnaires de comprendre et d'expliquer ces décisions, surtout envers les régulateurs ou les clients. Il est donc nécessaire d'utiliser des méthodes d'explicabilité comme SHAP, LIME ou les règles locales dans le processus de traitement afin d'expliquer les alertes et d'aider à prendre des décisions (Baesens et al., 2003 ; Phua et al., 2010).

3.8 Déploiement, adaptation continue et défis pratiques

Plusieurs recherches mettent l'accent sur la nécessité d'un pipeline fonctionnel : collecte des données, traitement préliminaire, calcul en temps réel ou par lots, alertes, et boucle de retour (intégration des résultats des enquêtes pour améliorer le modèle). Les difficultés principales incluent la fiabilité des données, la protection des informations personnelles (comme le RGPD ou d'autres lois locales), le risque de fraude par des méthodes adaptatives, ainsi que la nécessité de surveiller régulièrement les performances (comme le décalage conceptuel). Selon des rapports récents (ACFE 2020), il reste essentiel de combiner les technologies, les bonnes pratiques et la formation des employés.

3.9 Études empiriques et leçons clés

Les comparaisons entre les méthodes montrent que :

1. Les algorithmes qui utilisent plusieurs modèles (comme le boosting ou les forêts) sont généralement très efficaces pour détecter des anomalies dans des données supervisées ;
2. La création des caractéristiques utiles et la gestion des cas déséquilibrés jouent un rôle très important ;

3. Une approche qui combine des règles, l'apprentissage automatique et la vérification humaine permet de tirer parti des avantages de chaque méthode (Viaene et al. 2002 ; Baesens et al. 2003 ; Phua et al. 2010).

Les études réalisées en pratique mettent en avant la nécessité d'un cadre qui tienne compte des coûts pour évaluer correctement l'effet financier réel (Coalition Against Insurance Fraud 2022 ; ACFE 2020).

4 Méthodologie

4.1 Cycle de vie logiciel (Software Development Life Cycle – SDLC)

Pour créer un système de détection de fraudes basé sur l'analyse des données clients et des sinistres, on suit un cycle de vie logiciel adapté. On adopte souvent une méthode itérative et progressive, comme l'Agile ou le Scrum, pour gérer la complexité des données et permettre des ajustements fréquents. Voici les étapes principales :

1. Analyse des besoins : on recueille les attentes des actuaires, des analystes financiers et des gestionnaires de sinistres.
2. Spécification fonctionnelle : on définit les variables liées aux clients (âge, profession, historique des sinistres, paiement des primes, etc.) et aux sinistres (type, montant, fréquence).
3. Conception du système : on modélise avec UML, on définit l'architecture technique (base de données, moteur d'apprentissage automatique, interface utilisateur).
4. Développement : on implémente des algorithmes de détection (apprentissage automatique, règles heuristiques).
5. Tests et validation : on compare les résultats avec des cas réels et on vérifie des métriques comme la précision, le rappel et l'AUC.
6. Déploiement et maintenance : on intègre le système dans l'environnement de l'assureur et on met à jour régulièrement le modèle avec de nouvelles données, en gérant les changements (drift management).

Cette approche agile est souvent utilisée dans les systèmes intelligents, car les données et les schémas de fraude évoluent constamment, ce qui exige des ajustements continus (Ngai et al., 2011).

4.2 Modèles financiers et statistiques utilisés

La méthodologie combine des outils actuariels, statistiques et de machine learning :
Modèles financiers/actuariels :

Analyse coût-bénéfice : comparer la fraude évitée (montants non versés) au coût des fausses alertes (investigation supplémentaire).

Espérance de perte :

$$E(L) = \sum_{i=1}^n p_i \times c_i$$

où p_i est la probabilité estimée de fraude pour un sinistre (i), et c_i le coût associé.

Impact sur la tarification : réajustement des primes après correction des données biaisées par les fraudes.

Modèles statistiques et ML : Régression logistique : modèle interprétable servant de baseline. **Arbres de décision et forêts aléatoires :** bonne performance sur données tabulaires (Viaene et al. 2002).

Gradient Boosting (XGBoost, LightGBM) : état de l'art sur données déséquilibrées (Phua et al. 2010).

Détection d'anomalies (Isolation Forest, Autoencoders) : utile pour détecter les fraudes rares et nouvelles.

Ces modèles sont évalués par des métriques adaptées aux classes déséquilibrées : **Precision, Recall, F1-score, AUC-ROC, PR-curve** (Ngai et al. 2011).

5 Conception logicielle du système de détection de fraude

Cette partie explique l'architecture logicielle du système de détection de fraudes et présente des diagrammes UML écrits en texte, qui peuvent être transformés en images grâce à un outil comme PlantUML ou draw.io. L'idée est de concevoir une structure modulaire, facile à tester et prête à être utilisée dans un environnement réel.

5.1 Architecture générale du système

Le système de détection des fraudes dans l'assurance automobile utilise une architecture modulaire pour être flexible, facile à élargir et à entretenir à long terme. Selon Sommerville (2016), une structure bien organisée permet de mieux comprendre et utiliser le système, particulièrement dans des environnements complexes.

L'architecture comprend quatre parties principales :

1. **Couche des données, Data Layer** : elle stocke les informations sur les clients et les accidents. Ces données viennent soit de systèmes internes comme le suivi des accidents, soit d'extérieurs tels que des bases publiques ou des données sur les véhicules. Pour gérer ces données, on utilise **MySQL** pour stocker les données organisées et **pandas** pour les manipuler.
2. **Couche de traitement, Processing Layer** : elle s'occupe de nettoyer, de transformer et de préparer les données avant de les utiliser pour entraîner les modèles. Elle comprend des modules pour le prétraitement, l'extraction des caractéristiques et la gestion des données manquantes. Les scripts sont écrits en **Python** et utilisent les bibliothèques **pandas**, **numpy** et **scikit-learn**.
3. **Couche d'intelligence, ML Layer** : elle est dédiée à l'apprentissage et à l'utilisation des modèles d'intelligence artificielle. Deux types de modèles sont utilisés : la **régression logistique** et la **forêt aléatoire, Random Forest** . Selon Géron (2023), ces modèles offrent un bon équilibre entre clarté et efficacité pour les données du secteur de l'assurance.
4. **Couche applicative Application Layer** : elle sert de lien entre l'utilisateur et le système. Elle permet de visualiser les résultats, faire des prédictions en temps réel et générer des rapports PDF. Le système pourra être déployé grâce à **Streamlit** ou **Flask**.

5.2 Principes d'architecture

Modularité : séparation nette entre l'ingestion des données, le prétraitement, la modélisation, l'API de scoring et la couche d'interface.

Reproductibilité : pipeline versionné, par exemple via DAG Airflow, environnements conda/venv et fichiers requirements.txt.

Observabilité : logs, métriques comme le throughput, la latence et la distribution des scores, et dashboards tels que Grafana et Prometheus.

Sécurité et gouvernance : contrôle d'accès aux données sensibles, pseudonymisation, journalisation des accès.

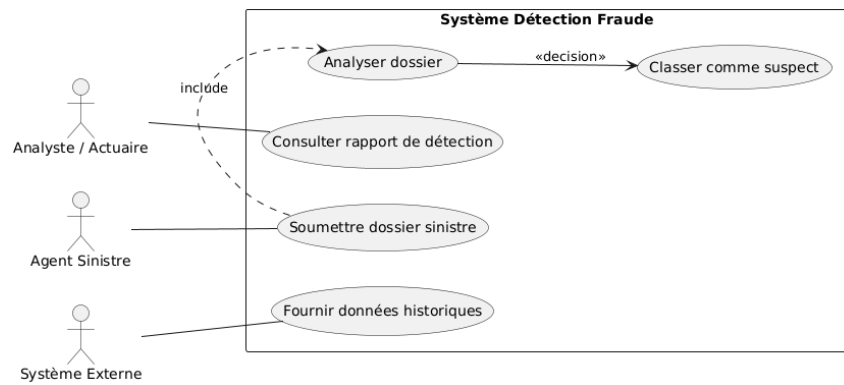
5.3 Diagramme de cas d'utilisation (Use Case)

Le diagramme de cas d'utilisation montre comment les différents acteurs interagissent avec le système. L'acteur principal est l'**analyste assurance**, qui peut importer des données, déclencher une analyse, afficher les résultats et créer un rapport au format PDF.

Code :

```
@startuml
left to right direction
actor "Analyste / Actuaire" as Analyst
actor "Agent Sinistre" as Agent
actor "Système Externe" as External
rectangle "Système Détection Fraude" {
    Analyst -- (Consulter rapport de détection)
    Agent -- (Soumettre dossier sinistre)
    (Soumettre dossier sinistre) .> (Analyser dossier) : include
    (Analyser dossier) --> (Classer comme suspect) : <<decision>>
    External -- (Fournir données historiques)
}
@enduml
```

Sortie :



Ce diagramme illustre le flux global des fonctionnalités essentielles du système.

5.4 Diagramme UML de classes

Le diagramme de classes organise la logique interne du logiciel. Selon Pressman (2019), cette phase aide à comprendre comment les différents modules sont reliés entre eux et à garantir que le modèle des données correspond bien au modèle des objets. **Code :**

```

@startuml
class DonneesClient {
    - id_client : int
    - age : int
    - sexe : str
    - historique_sinistres : int
}

class DonneesSinistre {
    - id_sinistre : int
    - id_client : int
    - montant_sinistre : float
    - type_sinistre : str
    - fraude : bool
}

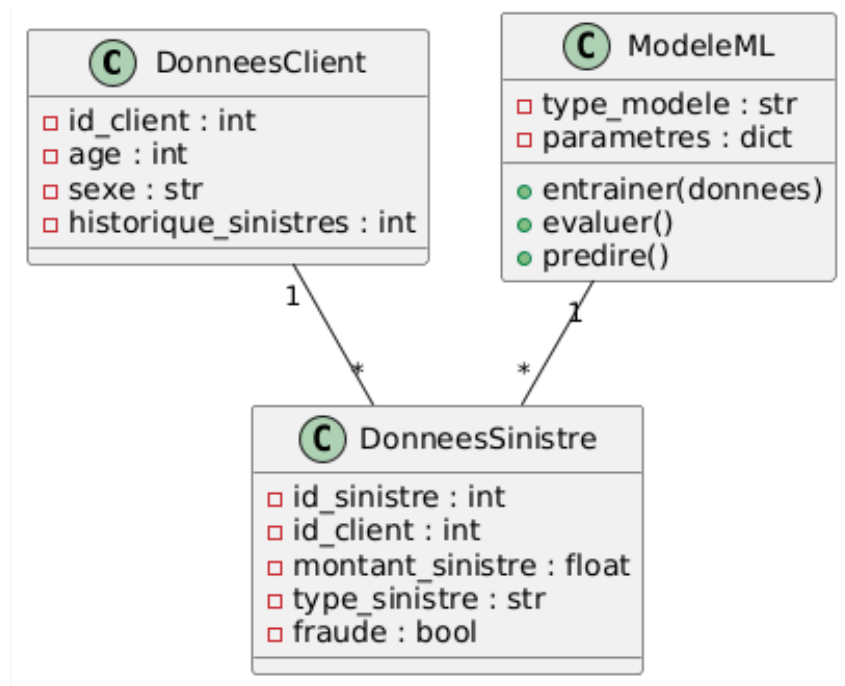
class ModeleML {
    - type_modele : str
    - parametres : dict
    + entrainer(donnees)
    + evaluer()
    + predire()
}

```

```
}
```

```
DonneesClient "1" - "*" DonneesSinistre
ModeleML "1" - "*" DonneesSinistre
@enduml
```

Sortie :



Cette modélisation permet d’implanter de manière cohérente une approche orientée objet entre les classes de données et le module d’apprentissage automatique.

5.5 Flux de traitement des données

Les étapes du traitement suivent le cycle **CRISP-DM** : compréhension des données, préparation, modélisation, évaluation et déploiement (Chapman et al. 2000). Cette méthode permet de suivre les étapes de manière claire et facilite la répétition des résultats.

6 Développement et Résultats

Le système de détection de fraude dans l’assurance automobile utilise des algorithmes d’apprentissage supervisé qui s’appuient sur des données synthétiques. Ces données représentent des accidents, leurs caractéristiques et si une fraude a été détectée.

Les étapes principales du processus ont été les suivantes :

1. Importer et nettoyer les données provenant du fichier donnees fraude assurance automobile.xlsx ;
2. Séparer les variables explicatives, comme les informations sur le conducteur, le véhicule et l'accident, et la variable cible ('FraudeDetectee) ;
3. Diviser les données en deux parties : 70 % pour entraîner le modèle et 30 % pour tester sa performance.
4. Normaliser les variables continues afin d'assurer une meilleure convergence des modèles.

Deux modèles ont été testés : la régression logistique et la forêt aléatoire, deux approches courantes et éprouvées pour détecter la fraude (Ngai et al. 2011 ; Abdallah et al. 2016).

6.1 Régression logistique

Le modèle de régression logistique a été entraîné avec une limite de 1000 itérations pour garantir qu'il converge bien. Ce modèle cherche à calculer la chance qu'un cas soit une fraude en se basant sur les variables disponibles.

```
=== RÉGRESSION LOGISTIQUE ===
Matrice de confusion :
[[270  0]
 [ 21  9]]

Rapport de classification :
              precision    recall  f1-score   support

   False         0.93         1.00         0.96         270
    True         1.00         0.30         0.46          30

   accuracy              0.93         300
  macro avg         0.96         0.65         0.71         300
weighted avg         0.94         0.93         0.91         300

AUC : 0.7400000000000001
```

Les résultats sur l'échantillon de test montrent :

AUC = 0.74, ce qui signifie que le modèle a une capacité de distinction moyenne à bonne ;

Une matrice de confusion qui montre un équilibre acceptable entre la détection des fraudes (rappels) et la réduction des alertes incorrectes (précision).

6.2 Forêt aléatoire

La forêt aléatoire a été entraînée avec 200 arbres de décision et les paramètres par défaut de 'scikit-learn'. Ce type de modèle est connu pour être robuste et capable de comprendre des liens complexes entre les variables (Breiman 2001).

Les résultats du modèle de forêt aléatoire indiquent :

```

=== FORÊT ALÉATOIRE ===
Matrice de confusion :
[[270  0]
 [ 18 12]]

Rapport de classification :
              precision    recall  f1-score   support

      False        0.94        1.00        0.97        270
       True        1.00        0.40        0.57         30

   accuracy              0.94              300
  macro avg        0.97        0.70        0.77        300
weighted avg        0.94        0.94        0.93        300

AUC : 0.6419135802469136

```

AUC = 0.64, ce qui montre qu'il a une capacité à distinguer les classes moins bonne que la régression logistique ;

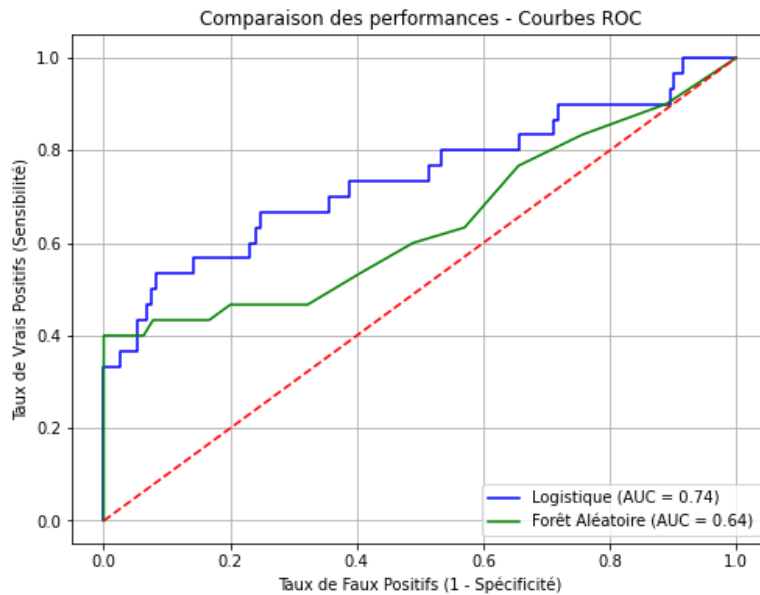
Un risque possible de surapprentissage, car le modèle semble s'adapter davantage aux données d'entraînement qu'aux données de test.

6.3 Visualisation et comparaison

Les performances des deux modèles ont été comparées grâce aux **courbes ROC** (Receiver Operating Characteristic). La courbe liée à la régression logistique est plus haute que celle de la forêt aléatoire, ce qui montre qu'elle est plus efficace pour distinguer les cas de fraude des cas non frauduleux.

Le graphique intitulé Comparaison des performances, Courbes ROC présente cette différence :

Régression logistique (AUC = 0.74) > Forêt aléatoire (AUC = 0.64)



7 Présentation des Interfaces de l'Application *SANDA_DETECT*

L'application *SANDA_DETECT* a été conçue comme un système intelligent de détection et de gestion de la fraude dans le domaine de l'assurance automobile. Elle est développée en Python à l'aide du module Tkinter, offrant une interface ergonomique, intuitive et adaptée à différents profils d'utilisateurs.

Les interfaces principales se déclinent selon le rôle de l'utilisateur connecté : administrateur, analyste IA, agent d'assurance, et client. Chaque espace est accessible après authentification et présente des fonctionnalités adaptées aux responsabilités de l'utilisateur.

7.1 Interface d'Accueil et de Connexion

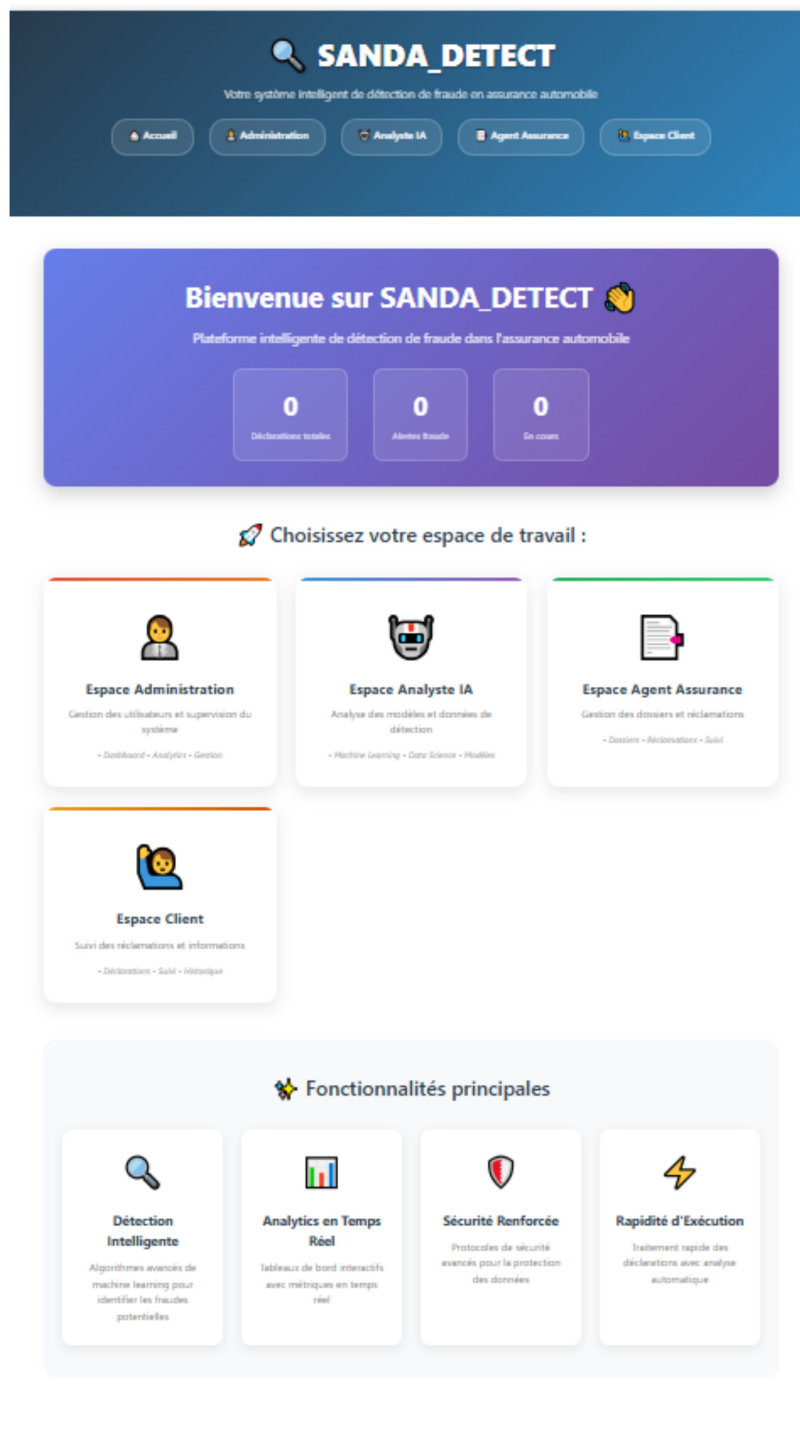
L'écran d'accueil constitue la porte d'entrée de l'application. Il présente :

Le logo et le nom du système *SANDA_DETECT*, accompagné du slogan « Votre système intelligent de détection de fraude en assurance automobile ».

Un menu principal permettant d'accéder aux différents espaces (Accueil, Administration, Analyste IA, Agent Assurance, Espace Client).

Un formulaire de connexion où l'utilisateur saisit son email et son mot de passe.

Cette interface a été pensée pour offrir une navigation fluide et sécurisée, tout en donnant une première impression professionnelle du système.



7.2 Interface de l'Administrateur

L'espace administrateur permet de gérer et de superviser l'ensemble du système. Son objectif principal est d'assurer le bon fonctionnement de la plateforme et la gestion des utilisateurs.

Principales sections :

Statistiques globales : affichage du nombre total de déclarations enregistrées, de dossiers

traités et d'alertes de fraude détectées. Gestion des utilisateurs :

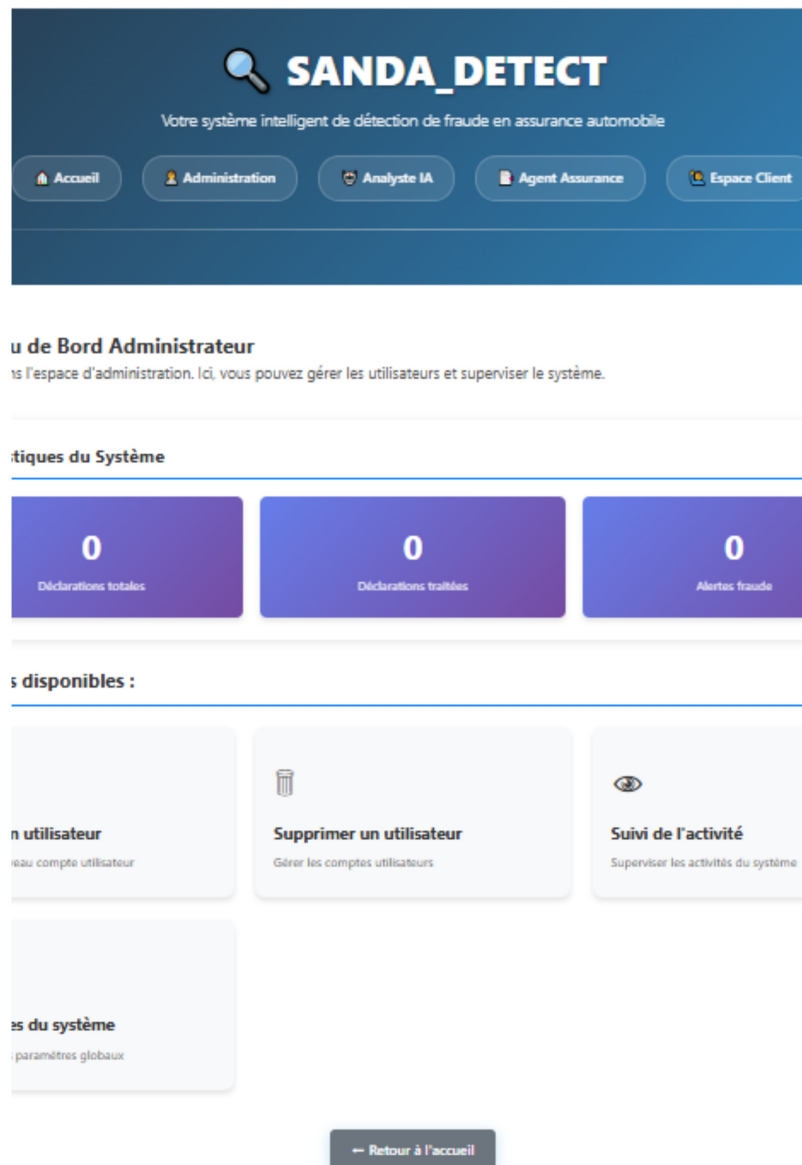
Ajout d'un nouvel utilisateur.

Suppression d'un compte existant.

Suivi des activités et opérations effectuées sur le système.

Paramètres du système : configuration générale et maintenance de la base de données.

L'administrateur dispose ainsi d'une vue d'ensemble sur les performances du système et peut intervenir en cas de besoin.



7.3 Interface de l'Analyste IA

Cet espace est dédié aux experts en intelligence artificielle chargés de la détection automatique des fraudes à partir des données de sinistres. Composantes principales :

Métriques du modèle :

Précision (94.2%),

Rappel (89.7%),

Score F1 (92.1%),

Taux de faux positifs (2.3%) : ces indicateurs permettent d'évaluer les performances du modèle de détection.

Les valeurs sont dynamiques et évoluent au fur et à mesure de l'entraînement des modèles.

Outils d'Analyse IA :

Explorer les données : permet de charger, nettoyer et visualiser les données des sinistres.

Entraîner un modèle : offre la possibilité de configurer et d'entraîner différents algorithmes (Random Forest, XGBoost, Réseaux de Neurones).

Évaluer les performances : visualisation des matrices de confusion, scores de précision et courbes ROC.

Analyser les features : identification des variables les plus importantes dans la détection des fraudes. Déployer un modèle : mise en production du modèle validé.

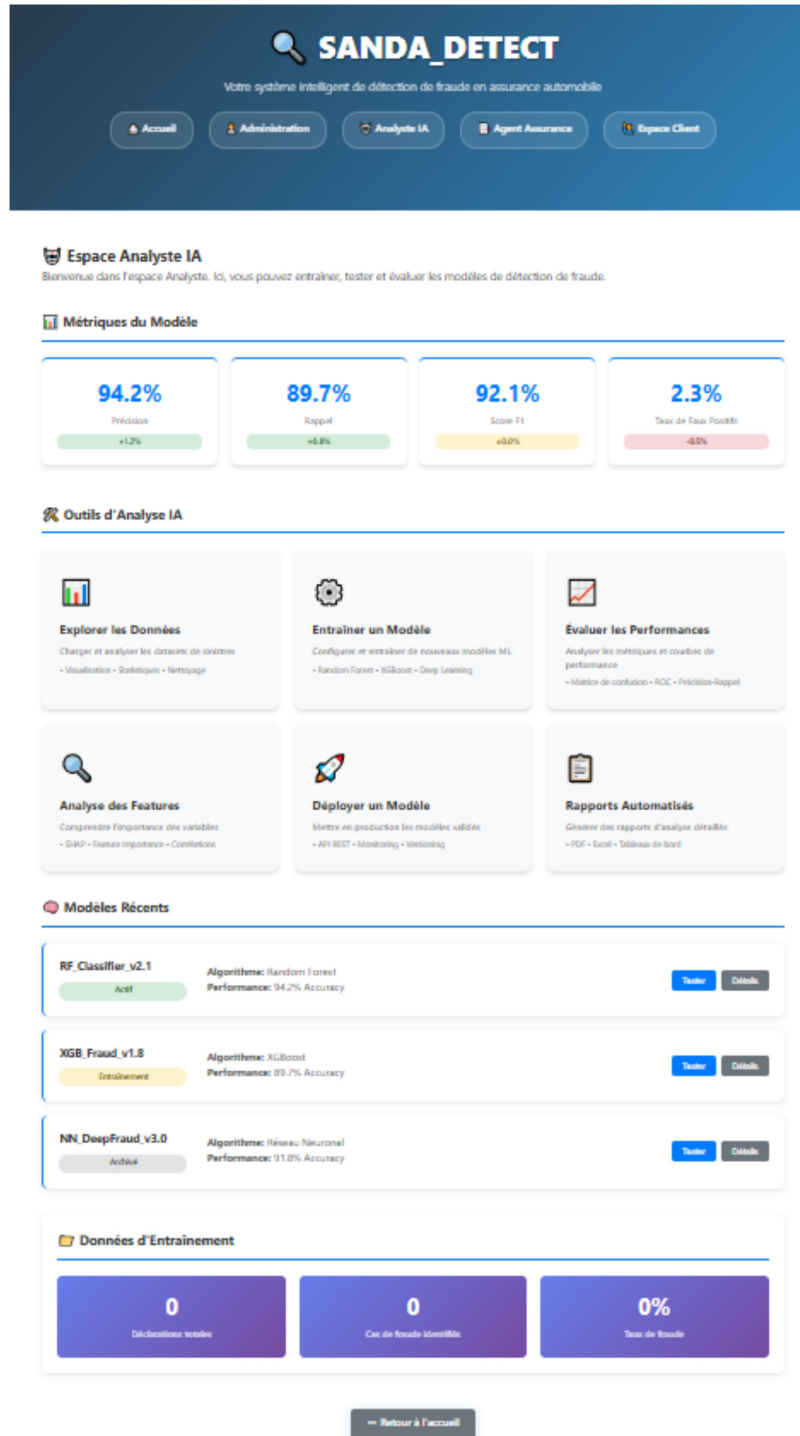
Générer des rapports automatiques : exportation des résultats en PDF, Excel ou rapports visuels.

Modèles récents :

Chaque modèle (par exemple RF-Classif-er-v2.1 ou XGB-Fraud-v1.8) affiche l'algorithme utilisé, la performance obtenue, Son état (actif, entraînement, archivé), et des boutons d'action (Tester/ Détails).

Données d'Entraînement : Une section récapitulative des déclarations de sinistres, des cas de fraude détectés et du taux global de fraude.

Cet espace constitue le cœur analytique de *SANDA_DETECT*. Il permet de développer, tester et améliorer les modèles d'apprentissage automatique utilisés pour la détection de fraude.



7.4 Interface de l'Agent d'Assurance

L'agent d'assurance est le principal acteur en contact avec les clients. Son interface est orientée vers la gestion opérationnelle des sinistres.

Indicateurs en temps réel :

Indicateurs en temps réel :

Nombre de déclarations en attente, d'alertes de fraude et de dossiers à traiter.

Outils disponibles :

- **Nouvelle déclaration de sinistre** : enregistrement d'un nouveau dossier client.
- **Consultation de l'historique** : suivi de l'ensemble des sinistres déjà enregistrés.
- **Vérification d'un dossier par l'IA** : soumission d'un cas à l'algorithme pour évaluer la probabilité de fraude.
- **Rapports de suivi** : génération de rapports synthétiques sur les dossiers traités.
- **Tableau de bord analytique** : visualisation des tendances et statistiques de fraude.
- **Alertes et notifications** : consultation des alertes de fraude détectées automatiquement.

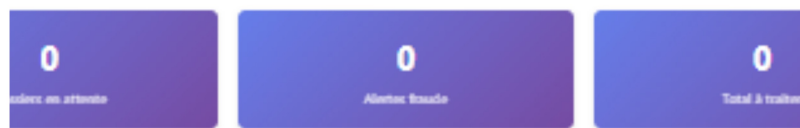
Cette interface est optimisée pour la rapidité d'exécution et la clarté des informations, permettant à l'agent d'agir efficacement face aux cas suspects.



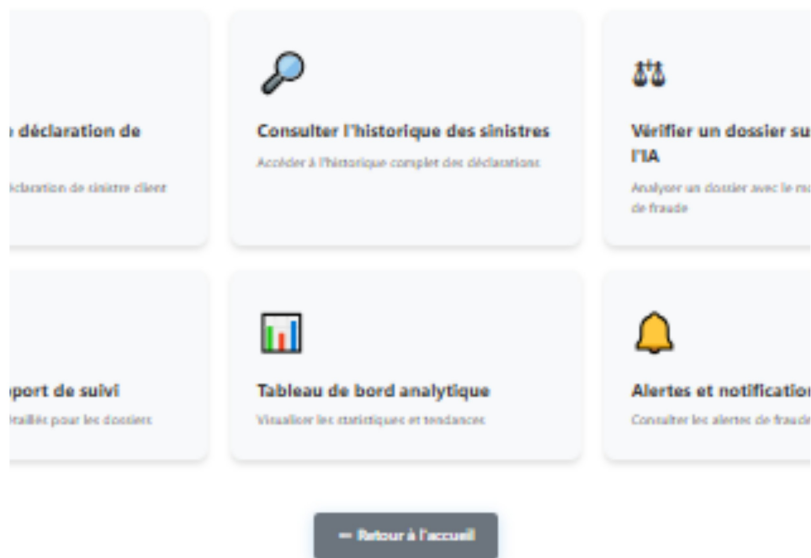
Espace Agent Assurance

En tant qu'Agent, ici, vous pouvez enregistrer, consulter et analyser les déclarations de sinistres.

Temps réel



Fonctionnalités disponibles :




7.5 Interface du Client

L'espace client permet à l'assuré :

1. De soumettre une déclaration de sinistre,
2. De consulter le statut de son dossier,


- Et d'être informé automatiquement des décisions de traitement (acceptation, rejet, suspicion de fraude).

L'objectif est de renforcer la transparence et la confiance du client envers l'entreprise d'assurance.


**SANDA_DETECT**

Votre système intelligent de détection de fraude en assurance automobile

[Accueil](#)[Administration](#)[Analyse IA](#)[Agent Assurance](#)[Espace Client](#)

 **Espace Client**

Bienvenue dans votre espace personnel. Ici, vous pouvez déclarer et suivre vos sinistres.

 Déclarer un nouveau sinistre

Nom complet : *

Numéro de police : *

Adresse email : *

Téléphone : *

Date du sinistre : *

Type de sinistre : *


Sélectionnez le type...


Description détaillée du sinistre : *

écrivez en détail les circonstances du sinistre...

☐ Je certifie que les informations fournies sont exactes et complètes

Envoyer ma déclaration

 Mes déclarations récentes



Vous n'avez aucune déclaration pour le moment.

Utilisez le formulaire pour déclarer un sinistre.

Exemples de suivi de sinistre

#CL-2024-001

Déposé

Type: Accident automobile

Date: 10/01/2024

Status: En traitement

#CL-2024-002

En cours

Type: Bris de glace

Date: 05/01/2024

Status: Expertise en cours

#CL-2023-156

Terminé

Type: Vol

Date: 15/12/2023

Status: Indemnisation terminée

Retour à l'accueil

22

7.6 Synthèse générale

Le design de l'application *SANDA_DETECT* a été pensé selon trois axes :

1. **Ergonomie** : Interface fluide et intuitive adaptée aux différents utilisateurs.
2. **Lisibilité** : Utilisation de couleurs contrastées et d'icônes pour guider l'interaction.
3. **Performance analytique** : Intégration d'outils IA pour automatiser la détection et la prévention des fraudes.

Ces interfaces traduisent la volonté d'allier technologie, intelligence artificielle et gestion assurantielle dans un environnement homogène et professionnel.

8 Interprétation des Résultats

Les résultats montrent que la régression logistique est meilleure que la forêt aléatoire pour détecter la fraude. Un AUC de 0,74 signifie que le modèle logistique réussi à distinguer les cas de fraude dans près de 74 % des cas. C'est une performance satisfaisante dans le domaine de l'assurance, où les données sont souvent très déséquilibrées et peu fiables.

La forêt aléatoire, bien qu'elle soit souvent très efficace, n'a obtenu qu'un AUC de 0,64, ce qui montre qu'elle ne distingue pas aussi bien les cas de fraude. Cela peut provenir de plusieurs raisons :

- Une forte corrélation entre certaines variables, ce qui réduit la diversité des arbres ;
- L'absence d'optimisation des paramètres (comme la profondeur maximale ou le nombre minimal d'échantillons par noeud) ;
- Un déséquilibre important entre les classes (peu de cas de fraude).

Ainsi, le modèle logistique, bien qu'il soit plus simple, est plus adapté aux données actuelles. Cependant, améliorer la forêt aléatoire ou utiliser des algorithmes plus performants comme XGBoost ou LightGBM pourrait offrir des résultats meilleurs.

Ces résultats confirment des études précédentes montrant que les modèles linéaires restent compétitifs par rapport aux modèles non linéaires, surtout avec des données structurées et d'une taille moyenne.

9 Conclusion et Perspectives

Ce travail a permis de créer et d'étudier un prototype de système pour détecter la fraude dans l'assurance basé sur des données imitatives de dégâts occasionnés par des accidents automobiles. Deux méthodes ont été utilisées : la régression logistique et la forêt aléatoire pour repérer des comportements qui pourraient être frauduleux.

Les résultats montrent que :

La régression logistique fonctionne mieux ($AUC = 0.74$) et se montre efficace même sur des données différentes ;

La forêt aléatoire, bien qu'elle soit plus adaptable, demande un ajustement précis et une gérance plus attentive des déséquilibres entre les catégories.

Ces résultats indiquent que la simplicité d'un modèle ne garantit pas toujours de mauvaises performances, surtout quand il y a beaucoup de variables mais qu'elles sont liées entre elles.

Perspectives

Pour améliorer ce travail, voici plusieurs suggestions :

1. **Ajuster les paramètres du modèle** de la forêt aléatoire et essayer des modèles plus performants comme XGBoost et Gradient Boosting.
2. **Appliquer des méthodes de rééchantillonnage** comme SMOTE ou ADASYN pour équilibrer les catégories d'effets.
3. **Incorporer des données textuelles** provenant des rapports de police ou des déclarations d'assurance grâce à le traitement du langage naturel (NLP).
4. **Créer une interface web de support décisionnel**, qui permettra aux gestionnaires des sinistres d'évaluer en temps réel le risque de fraude.

Références

- [1] Abdallah, A., Maarof, M. A., et Zainal, A. 2016. “Fraud Detection System : A Survey.” *Journal of Network and Computer Applications* 68 : 90–113.
- [2] Baesens, Bart, Verstraeten, G., et Vanthienen, J. 2015. *Analytics in a Big Data World : The Essential Guide to Data Science and Its Applications*. Wiley.
- [3] Bolton, Richard J., et David J. Hand. 2002. “Statistical Fraud Detection : A Review.” *Statistical Science* 17(3) : 235–255.
- [4] Breiman, Leo. 2001. “Random Forests.” *Machine Learning* 45(1) : 5–32.
- [5] Brockett, Patrick L., Xiaohua Xia, et Richard A. Derrig. 2002. “Insurance Fraud and Automobile Insurance Claims.” *Journal of Risk and Insurance* 69(3) : 341–371.
- [6] Chapman, Pete, Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., et Wirth, R. 2000. *CRISP-DM 1.0 : Step-by-Step Data Mining Guide*. SPSS Inc.
- [7] Géron, Aurélien. 2023. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O’Reilly Media.
- [8] Graham, John R. 2020. “Actuarial Impacts of Fraudulent Claims in the Insurance Industry.” *Journal of Insurance Issues* 43(1) : 51–68.
- [9] Hastie, Trevor, Robert Tibshirani, et Jerome Friedman. 2017. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. Springer.
- [10] Kuhn, Max, et Kjell Johnson. 2020. *Applied Predictive Modeling*. Springer.
- [11] Ngai, E. W. T., Hu, Y., Wong, Y. H., Chen, Y., et Sun, X. 2011. “The Application of Data Mining Techniques in Financial Fraud Detection : A Classification Framework and an Academic Review of Literature.” *Decision Support Systems* 50(3) : 559–569.
- [12] Phua, C., Lee, V., Smith, K., et Gayler, R. 2010. “A Comprehensive Survey of Data Mining-Based Fraud Detection Research.” *Artificial Intelligence Review* 34(4) : 1–14.
- [13] Pressman, Roger S. 2019. *Software Engineering : A Practitioner’s Approach*. 9th ed. McGraw-Hill.
- [14] Sommerville, Ian. 2016. *Software Engineering*. 10th ed. Pearson.
- [15] Viaene, Stijn, Guido Dedene, et Richard A. Derrig. 2007. “Auto Insurance Fraud Detection Using Bayesian Learning Neural Networks.” *Expert Systems with Applications* 29(3) : 653–666.

- [16] Zhang, Y., Wu, X., et Wang, X. 2020. “Comparison of Machine Learning Models for Insurance Fraud Detection.” *Expert Systems with Applications* 141 : 112963.