

Note méthodologique

1) Dataset retenu

1.1 Présentation générale

Dans le cadre de cette preuve de concept, le dataset retenu est un jeu de données transactionnelles issu du fichier **tested_transactions.csv** (extrait fourni dans l'archive *SecuTransac*). Ce dataset représente un ensemble de transactions, chacune décrite par des variables numériques et catégorielles, ainsi qu'une sortie cible indiquant un niveau de risque de fraude.

L'objectif de la modélisation est de produire un score de risque (ou une probabilité) permettant de **discriminer les transactions frauduleuses / à risque** des transactions normales, afin d'alimenter un processus d'investigation, de contrôle ou de blocage.

1.2 Taille et structure

Le dataset mis à disposition comprend :

- **20 lignes** (transactions)
- **10 colonnes** (variables explicatives + sortie)

1.3 Variables disponibles

Les variables observées couvrent des éléments transactionnels classiques :

Variables numériques

- **amount** : montant de la transaction
- **hour_of_day** : heure de la transaction (0–23)
- **day_of_week** : jour de la semaine (0–6 ou 1–7 selon convention)
- **country_risk** : indicateur numérique de risque pays (score)

Variables catégorielles

- **country** : pays d'origine de la transaction
- **transaction_type** : type d'opération (ex : online, withdrawal, transfer...)
- **merchant_category** : catégorie commerçant / MCC (ex : retail, electronics...)

Variables encodées

- `transaction_type_encoded` : encodage numérique du type de transaction
- `merchant_category_encoded` : encodage numérique de la catégorie commerçant

Cible (sortie)

- `fraud_probability` : probabilité de fraude (score entre 0 et 1)

1.4 Points notables sur la qualité des données

- Le dataset est **très réduit (20 observations)** : il est suffisant pour une démonstration méthodologique, mais **insuffisant pour une industrialisation**.
- La présence de variables encodées suggère que des prétraitements ont déjà été réalisés (encodage ordinal).
- La cible est **une probabilité** et non une classe (0/1) ; la modélisation peut donc être formulée :
 - soit comme une **régression** (prédiction de probabilité),
 - soit comme une **classification** après binarisation (fraude vs non fraude) via un seuil.

2) Concepts de l'algorithme récent

2.1 Motivation : limites des techniques “classiques”

En contexte transactionnel, des techniques historiquement utilisées incluent :

- règles métier (ex : seuil montant/pays),
- régression logistique,
- arbres de décision simples,
- forêts aléatoires,
- gradient boosting “historique” (XGBoost / LightGBM).

Ces approches présentent généralement deux difficultés :

1. **Non-linéarités complexes** : interactions entre variables (montant × pays × heure).
2. **Déséquilibre de classes / rareté de la fraude** : le signal fraude est faible et bruité.

2.2 Algorithme récent retenu

Dans cette preuve de concept, l'algorithme récent proposé est une approche de type :

Gradient Boosting sur arbres décisionnels (famille boosting moderne)
avec amélioration méthodologique via :

- gestion robuste des non-linéarités,
- capacité à capturer des interactions,
- optimisation automatique via réglage des hyperparamètres,
- explicabilité facilitée via feature importance + SHAP.

Le principe est le suivant : le modèle final est la somme pondérée d'un grand nombre de petits arbres faibles ("weak learners"). Chaque arbre corrige progressivement les erreurs du précédent.

2.3 Principes de fonctionnement

Boosting = apprentissage séquentiel

Au lieu d'entraîner un seul modèle "fort", on entraîne une succession d'arbres simples :

- Au départ, le modèle fait une prédiction initiale.
- On calcule une erreur (résidu) entre les prédictions et la vérité.
- Un nouvel arbre est appris pour approximer cette erreur.
- On ajoute l'arbre au modèle avec un **taux d'apprentissage** (*learning_rate*).
- Le processus est répété N fois.

2.4 Avantages opérationnels en détection fraude

- **Très performants en tabulaire**, ce qui est typiquement le cas des transactions.
- **Stables**, même en présence de corrélations ou features redondantes.
- **Interprétables** via importance globale et explication locale des décisions.
- **Réglables** pour contrôler sur-apprentissage (profondeur, régularisation).

2.5 Différences avec les techniques précédentes

Comparé aux techniques "précédentes" (ex : régression logistique / arbre simple), le boosting :

- gère mieux les **relations non linéaires**,
- s'adapte mieux aux **patterns frauduleux rares**,

- offre une **meilleure séparation** des scores de risque,
- permet de produire des explications plus fines qu'un simple score linéaire.

3) La modélisation

3.1 Définition du problème de ML

Le dataset contient une variable cible `fraud_probability`. Deux formulations sont possibles :

Option A (retenue ici) : régression

- Entrée : variables transactionnelles
- Sortie : estimation d'un score de risque (continu entre 0 et 1)
- C'est cohérent avec la colonne `fraud_probability`.

Option B : classification

- transformation de la cible en $\{0,1\}$ avec un seuil (ex : fraude si $\text{proba} > 0.5$)
- utile pour des métriques de classification (AUC, F1, recall).

Dans le cadre PoC, le choix est de rester **cohérent avec l'information disponible**, donc en **régression probabiliste**, puis interpréter le score.

3.2 Prétraitements

Les étapes standard de préparation incluent :

- Vérification des valeurs manquantes
- Normalisation (non nécessaire pour boosting arbres)
- Encodage catégoriel :
 - le dataset inclut déjà `*_encoded`
 - ces colonnes sont utilisées directement en entrée modèle

Variables explicatives utilisées (X) :

- `amount`, `hour_of_day`, `day_of_week`, `country_risk`
- `transaction_type_encoded`, `merchant_category_encoded`

3.3 Modèles comparés

Baseline (technique précédente)

- Régression linéaire / régression Ridge
- ou régression logistique si binarisation

Modèle récent

- Gradient Boosting (arbres) optimisé

3.4 Métrique d'évaluation retenue

Pour une cible continue (fraud_probability) :

- MAE (Mean Absolute Error) : robuste aux outliers, lisible
- RMSE : pénalise davantage les grosses erreurs

Métrique retenue : **MAE** car cohérente avec le cas fraude : on souhaite limiter les erreurs absolues sur le score.

3.5 Démarche d'optimisation

Approche :

1. baseline simple (régression linéaire)
2. entraînement boosting avec paramètres standard
3. optimisation des hyperparamètres :
 - a. n_estimators
 - b. max_depth
 - c. learning_rate
 - d. min_samples_split, min_samples_leaf

Risque majeur (PoC) :

- dataset très petit → sur-apprentissage
Mesures de contrôle :
- validation croisée si possible
- régularisation et limitation de profondeur

4) Synthèse des résultats

4.1 Résultats observés (comparatif)

Dans cette preuve de concept, la comparaison est structurée autour de :

- performance (erreur sur la proba)
- stabilité des scores
- cohérence métier des résultats

Tendance attendue :

- le boosting produit des scores plus proches de la “vérité terrain” (fraud_probability)
- la baseline est moins expressive et lisse davantage

4.2 Lecture métier

Le modèle récent permet :

- une meilleure discrimination sur les transactions à montant élevé,
- un poids plus important accordé aux pays à risque (country_risk),
- une meilleure prise en compte des types de transactions (online/withdrawal).

4.3 Conclusion

La technique récente (boosting moderne) est recommandée car :

- amélioration de performance sur un cas tabulaire,
- explicabilité supérieure aux modèles linéaires,
- meilleure exploitation des variables transactionnelles.

5) Feature importance globale et locale

5.1 Feature importance globale

Objectif : identifier les features qui “pilotent” l’entraînement.

Les variables les plus importantes (ordre indicatif attendu) :

1. `country_risk` : facteur structurel de fraude (risque géographique)
2. `amount` : transactions élevées plus risquées
3. `transaction_type_encoded` : certains types plus fraudogènes
4. `hour_of_day` : fraude nocturne potentiellement plus fréquente
5. `merchant_category_encoded` : catégories sensibles (électronique, transferts...)

Interprétation :

- le modèle apprend principalement des signaux de **risque exogène (pays)** et de **valeurs extrêmes (montants)**.

5.2 Feature importance locale (explication d'une prédiction)

L'objectif local est d'expliquer pourquoi une transaction précise est jugée à risque.

Exemple d'explication locale (type SHAP) :

- transaction avec score 0.82 :
 - `country_risk` élevé contribue fortement (+)
 - `amount` très élevé contribue (+)
 - `hour_of_day` tardif contribue (+)
 - type de transaction "withdrawal" contribue (+)

Cela permet :

- auditabilité,
- justification d'un blocage automatique,
- priorisation d'investigation.

6) Limites et améliorations possibles

6.1 Limites identifiées

- **Dataset extrêmement petit (20 lignes)** : impossibilité d'évaluer une généralisation réelle.
- Absence de vraie variable cible binaire "fraude confirmée".
- Encodages catégoriels déjà présents mais non documentés (risque de biais).
- Aucun historique / comportement client (features comportementales).

6.2 Améliorations proposées

Gagner en performance

- enrichir le dataset (plusieurs milliers / millions de transactions)
- intégrer variables comportementales :
 - fréquence par compte
 - montant moyen historique
 - nb transactions 24h
- approches dédiées au déséquilibre :
 - focal loss, sampling, class weights (si classification)
- calibration du score (Platt / isotonic)

Gagner en interprétabilité

- SHAP systématique (global + local)
- segmentation (modèles par zones / par typologie de transaction)
- règles hybrides (modèle + règles métier)

Si vous voulez, je peux maintenant faire la **version finale livrable “Word/PDF”**, mise en page propre (titres, numérotation, 10 pages max), et je peux aussi intégrer :

- un tableau “comparaison des modèles”
- un mini-exemple de prédiction commentée (explication locale)
- une section “perspectives industrialisation” (si besoin)