Résultats modèles de détection des fraudes pour identifier les transactions potentiellement frauduleuses.

- Mélanger (ou shuffler) un dataframe au début peut aider à éviter certains biais dans la prédiction :

    - **Préserve la distribution statistique :**
      En mélangeant les données, on garantit que les répartitions des catégories (ou des valeurs cibles) sont similaires dans les jeux d'entraînement et de test. Cela évite que l'un des jeux soit biaisé ou dominé par une catégorie particulière.

    - **Prévient les mauvaises performances sur les données non vues :**
      Si les données sont ordonnées et non mélangées, il y a un risque que le modèle ne soit pas exposé à la variété complète des cas possibles lors de l'entraînement, ce qui peut dégrader ses performances sur des nouvelles données.

- Je ne **normalise pas** les **données numériques** avec **Random Forest** car cet algorithme est basé sur des arbres de décision, qui sont insensibles à l'échelle des variables et fonctionnent uniquement sur des seuils de partitionnement des données.

- **SMOTE** est appliqué uniquement sur les données d'entraînement, car nous souhaitons que les données de test reflètent une situation réelle où la classe frauduleuse est minoritaire. Cela permet de confronter le modèle à un jeu de données équilibré lors de l'entraînement tout en gardant les données de test intactes pour évaluer les performances du modèle dans des conditions réelles, où les données sont naturellement déséquilibrées.

- J'ai voulu utiliser **SMOTE** sur les données d'entraînement dans mon jeu de données déséquilibré pour augmenter la représentation des classes minoritaires, afin d'améliorer la capacité du modèle à détecter ces classes.

- Les résultats montrent que l'utilisation de SMOTE améliore le rappel (0.7563 contre 0.7053 sans SMOTE), ce qui signifie que le modèle capture mieux les cas minoritaires. Cependant, sans SMOTE, la précision est nettement meilleure (0.9765 contre 0.8124 avec SMOTE), ce qui réduit les faux positifs. Les F1-Scores sont globalement similaires dans les deux cas, avec un léger avantage pour la méthode sans SMOTE. Étant donné que nous sommes dans un problème de détection de fraude, où il est crucial de minimiser les faux négatifs pour ne pas manquer des cas frauduleux, le rappel est prioritaire. Par conséquent, nous choisissons d'utiliser SMOTE pour équilibrer les classes et améliorer la capacité du modèle à détecter les fraudes.

- XGBoost est plus rapide que Random Forest grâce à son optimisation avancée, notamment l'élagage des arbres, la gestion efficace de la mémoire, et l'utilisation du

parallélisme pour accélérer l'entraînement, ce qui lui permet de converger plus rapidement vers un modèle performant.

- Nous préférons utiliser XGBoost sans SMOTE, car bien que SMOTE améliore le rappel, il entraîne une forte diminution de la précision, ce qui génère trop de faux positifs. Dans un problème de détection de fraude, il est crucial de maintenir une bonne précision pour éviter les alertes inutiles, et l'approche sans SMOTE offre un meilleur compromis entre précision et rappel.

- Les résultats montrent clairement que **XGBoost** surpasse **Random Forest** en termes de performance, avec un **F1-score** plus élevé (0.8926 contre 0.7833), ainsi qu'un **rappel** supérieur (0.8357 contre 0.7563), ce qui est crucial pour un problème de détection de fraude. De plus, XGBoost présente l'avantage supplémentaire d'être **beaucoup plus rapide lors de l'entraînement**, grâce à son algorithme optimisé et à l'utilisation du parallélisme, ce qui le rend non seulement plus performant, mais aussi plus efficace en termes de temps de calcul.

- Avec **LightGBM**, il n'est généralement pas nécessaire de normaliser les données numériques, car ce modèle, basé sur des arbres de décision, est insensible à l'échelle des données.

- Il est important de **normaliser les données** pour un modèle de **régression logistique**, car ce modèle est sensible à l'échelle des variables. Si les caractéristiques ont des échelles très différentes, certaines peuvent dominer le modèle, ce qui peut entraîner des biais dans les coefficients estimés et affecter la performance du modèle. La normalisation permet donc d'assurer que toutes les variables contribuent de manière égale à la prédiction.

- Il faut **normaliser les données** avec un **SVM** car ce modèle est sensible à l'échelle des variables, et des caractéristiques avec des échelles différentes peuvent affecter la performance et l'efficacité du noyau (en particulier pour les noyaux comme RBF).

Modèle	**Pré-traitements**	**Balance des classes**	**Cross-Validation (3 folds)**	**Test**
Random Forest	**Variables catégorielles :** ● One-Hot Encoding si moins de 20-30 catégories. ● Label Encoding si 2 catégories.	**Oui** : Smote Appliqué sur l'ensemble de Train (Sur-échantillonnage de la classe minoritaire)	**Fold 1** Precision: 0.8032, Recall: 0.7553, F1-Score: 0.7785 **Fold 2** Precision: 0.8141, Recall: 0.7451,	Precision: **0.8510** Recall: **0.7872** F1-Score: **0.8179** AUC-ROC (Test Set): **0.9936**

	• Target Encoding pour le reste. **Variables numériques :** • Pas de normalisation.		F1-Score: 0.7781 **Fold 3** Precision: 0.8200, Recall: 0.7685, F1-Score: 0.7934 **--- Résultats globaux sur tous les folds —** Precision: **0.8124** (± 0.0070) Recall: **0.7563** (± 0.0096) F1_score: **0.7833** (± 0.0072)	
Random Forest	**Variables catégorielles :** • One-Hot Encoding si moins de 20-30 catégories. • Label Encoding si 2 catégories. • Target Encoding pour le reste. **Variables numériques :** • Pas de normalisation.	Non	**Fold 1** Precision: **0.9768**, Recall: 0.6922, F1-Score: 0.8102 **Fold 2** Precision: 0.9739, Recall: 0.7012, F1-Score: 0.8153 **Fold 3** Precision: 0.9787, Recall: 0.7226, F1-Score: 0.8314 --- **Résultats globaux sur tous les folds** — Precision: **0.9765** (± 0.0020) Recall: **0.7053** (± 0.0128) F1_score: **0.8190** (± 0.0090)	Precision: **0.9853** Recall: 0.7353 F1-Score: **0.8421** AUC-ROC (Test Set): **0.9878**
XGBoost	**Variables catégorielles :** • One-Hot Encoding si moins de 20-30	non	**Fold 1** Precision: 0.9535, Recall: 0.8278, F1-Score: 0.8862	Precision: **0.9686** Recall: **0.8477** F1-Score: **0.9041**

	catégories. ● Label Encoding si 2 catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Pas de normalisation		**Fold 2** Precision: 0.9595, Recall: 0.8447, F1-Score: 0.8984 **Fold 3** Precision: 0.9607, Recall: 0.8344, F1-Score: 0.8931 **--- Résultats globaux sur tous les folds —** Precision: 0.9579 (± 0.0032) Recall: 0.8357 (± 0.0069) F1_score: 0.8926 (± 0.0050)	AUC-ROC (Test Set): 0.9991
XGBoost	**Variables catégorielles :** ● One-Hot Encoding si moins de 20-30 catégories. ● Label Encoding si 2 catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Pas de normalisation	**Oui** : Smote Appliqué sur l'ensemble de Train (Sur-échantillonnage de la classe minoritaire)	**Fold 1** Precision: 0.4390, Recall: 0.8706, F1-Score: 0.5837 **Fold 2** Precision: 0.4555, Recall: 0.8804, F1-Score: 0.6003 **Fold 3** Precision: 0.4403, Recall: 0.8862, F1-Score: 0.5883 **--- Résultats globaux sur tous les folds —** Precision: 0.4449 (± 0.0075) Recall: 0.8791 (± 0.0065) F1_score: 0.5908 (± 0.0070)	Precision: 0.4230 Recall: 0.9056 F1-Score: 0.5767 AUC-ROC (Test Set): 0.9948
LightGBM	**Variables catégorielles :** ● One-Hot Encoding	**Oui** : Smote Appliqué sur l'ensemble de Train	Fold 1 Precision: 0.2893, Recall: 0.8898, F1-Score: 0.4367	Precision: 0.2938 Recall: 0.9036 F1-Score:

	si moins de 20-30 catégories. ● Label Encoding si 2 catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Pas de normalisation	(Sur-échantillonnage de la classe minoritaire)	Fold 2 Precision: 0.2935, Recall: 0.8906, F1-Score: 0.4415 Fold 3 Precision: 0.3002, Recall: 0.9051, F1-Score: 0.4508 **--- Résultats globaux sur tous les folds ---** Precision: 0.2943 (± 0.0045) Recall: 0.8952 (± 0.0070) F1_score: 0.4430 (± 0.0059)	0.4434 AUC-ROC (Test Set): 0.9945
LightGBM	**Variables catégorielles :** ● One-Hot Encoding si moins de 20-30 catégories. ● Label Encoding si 2 catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Pas de normalisation	Non	Fold 1 Precision: 0.6565, Recall: 0.6357, F1-Score: 0.6459 Fold 2 Precision: 0.7252, Recall: 0.7525, F1-Score: 0.7386 Fold 3 Precision: 0.6737, Recall: 0.6155, F1-Score: 0.6433 --- Résultats globaux sur tous les folds — Precision: 0.6852 (± 0.0292) Recall: 0.6679 (± 0.0604) F1_score: 0.6760 (± 0.0443)	Precision: 0.7203 Recall: 0.6803 F1-Score: 0.6997 AUC-ROC (Test Set): 0.8638
Regression Logistic	**Variables catégorielles :** ● One-Hot Encoding si moins de 20-30	non	Fold 1 Precision: 0.6564, Recall: 0.1169, F1-Score: 0.1984	Precision: 0.6619 Recall: 0.1164 F1-Score: 0.1980

	catégories. ● Label Encoding si 2 catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Standardisation		Fold 2 Precision: 0.6197, Recall: 0.1086, F1-Score: 0.1849 Fold 3 Precision: 0.6927, Recall: 0.1220, F1-Score: 0.2075 **--- Résultats globaux sur tous les folds —** Precision: 0.6562 (± 0.0298) Recall: 0.1158 (± 0.0055) F1_score: 0.1969 (± 0.0093)	AUC-ROC (Test Set): 0.8482
Regression Logistic	**Variables catégorielles :** ● One-Hot Encoding si moins de 20-30 catégories. ● Label Encoding si 2 catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Standardisation	**Oui** : Smote Appliqué sur l'ensemble de Train (Sur-échantillonnage de la classe minoritaire)	Fold 1 Precision: 0.0416, Recall: 0.8192, F1-Score: 0.0791 Fold 2 Precision: 0.0430, Recall: 0.8008, F1-Score: 0.0815 Fold 3 Precision: 0.0421, Recall: 0.8156, F1-Score: 0.0800 --- Résultats globaux sur tous les folds --- Precision: 0.0422 (± 0.0006) Recall: 0.8119 (± 0.0080) F1_score: 0.0802 (± 0.0010)	--- Évaluation sur l'ensemble de test --- Precision: 0.0437 Recall: 0.8037 F1-Score: 0.0828 AUC-ROC (Test Set): 0.9292
SVM	**Variables catégorielles :** ● One-Hot Encoding si moins de 20-30 catégories. ● Label Encoding si 2	**Oui** : Smote Appliqué sur l'ensemble de Train (Sur-échantillonnage de la classe		

	catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Standardisation	minoritaire)		

. Évaluation du modèle :

J'utilise le **score F1**, la **précision** et le **rappel** comme métriques d'évaluation dans mon jeu de données déséquilibré pour la détection de fraude, car le **score F1** permet de trouver un équilibre entre la précision et le rappel, crucial dans ce contexte où les faux négatifs et les faux positifs ont un coût élevé, tandis que la **précision** évalue la qualité des prédictions positives (éviter les faux positifs), et le **rappel** mesure la capacité du modèle à identifier correctement les fraudes réelles (éviter les faux négatifs).

Optimisation des hyperparamètres :

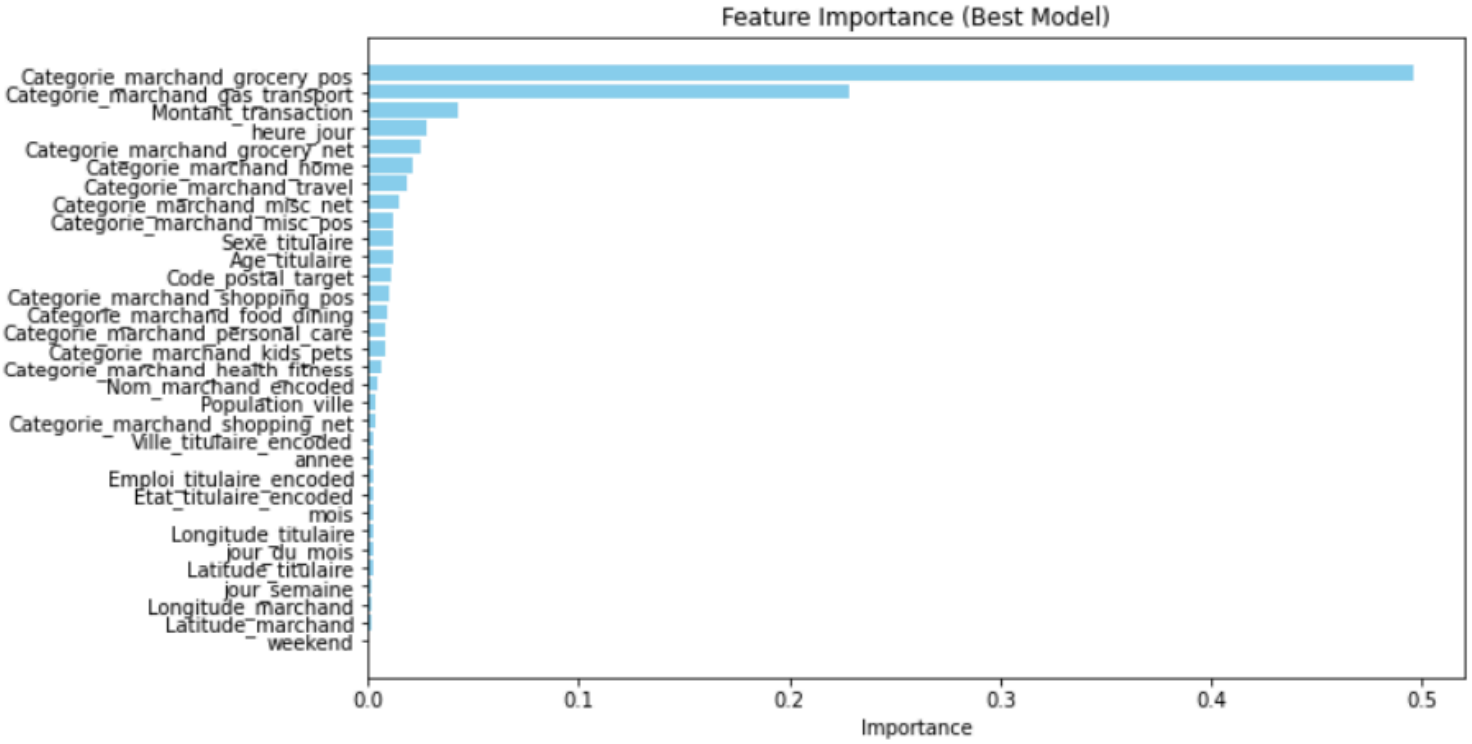
Modèle	Pré-traitements	Balance des classes	Grid Search Cv	Test
XGBoost	**Variables catégorielles :** ● One-Hot Encoding si moins de 20-30 catégories. ● Label Encoding si 2 catégories. ● Target Encoding pour le reste. **Variables numériques :** ● Pas de normalisation	non	Fitting 5 folds for each of 108 candidates, totalling 540 fits Meilleurs hyperparamètres : {'colsample_bytree': 1.0, 'learning_rate': 0.2, 'max_depth': 9, 'n_estimators': 200, 'subsample': 0.8}	Precision: **0.9779** Recall: **0.8621** F1-Score: **0.9164** AUC-ROC (Test Set): 0.9991

Précision (0.9779) : Très élevée, ce qui signifie que la majorité des fraudes détectées sont réellement des fraudes (peu de faux positifs). C'est important pour éviter de signaler des transactions légitimes comme frauduleuses.

Rappel (0.8621) : Un rappel solide, ce qui indique que le modèle capture une grande proportion des fraudes réelles (même s'il en manque encore quelques-unes). C'est acceptable tant que le rappel reste cohérent avec les besoins métier.

F1-Score (0.9164) : Bon équilibre entre précision et rappel, ce qui en fait une métrique pertinente pour ce type de problème déséquilibré.

Importance des Features :



Feature	Importance
Categorie_marchand_grocery_pos	0.495616
Categorie_marchand_gas_transport	0.228041
Montant_transaction	0.043144
heure_jour	0.028031
Categorie_marchand_grocery_net	0.025371

Categorie_marchand_home	0.021430
Categorie_marchand_travel	0.018810
Categorie_marchand_misc_net	0.014631
Categorie_marchand_misc_pos	0.012022
Sexe_titulaire	0.011776

Categorie_marchand_grocery_pos (0.495616)

- Interprétation : Cette caractéristique a la plus grande importance, avec une valeur de 49,56 %. Elle représente les transactions dans la catégorie épicerie effectuées en magasin physique (POS = point of sale).

- Explication pour les équipes métiers : Il semble que les transactions effectuées dans des magasins d'épicerie (en particulier ceux utilisant des systèmes de paiement en magasin) soient fortement corrélées aux fraudes. Cela pourrait signifier que certaines fraudes se produisent dans des lieux physiques spécifiques, comme des épiceries, ou que des comportements frauduleux sont plus fréquents dans ces types de transactions.

2. Categorie_marchand_gas_transport (0.228041)

- Interprétation : Les transactions dans la catégorie carburant/transport représentent 22,8 % de l'importance.
- Explication pour les équipes métiers : Les fraudes dans ce secteur peuvent être fréquentes en raison de la nature des transactions, notamment avec des montants relativement faibles mais fréquents. Les transactions de transport, notamment dans le secteur des carburants, sont souvent ciblées par des fraudeurs qui cherchent à effectuer plusieurs petites transactions pour éviter les déclenchements automatiques de systèmes de détection de fraude.

3. Montant_transaction (0.043144)

- Interprétation : Le montant de la transaction est moins important dans le modèle, mais il reste un facteur qui influence la détection de fraude.

- Explication pour les équipes métiers : Bien que le montant de la transaction soit une caractéristique importante pour détecter des fraudes dans certains systèmes, il semble que, dans ce cas, des éléments comme la catégorie de marchand jouent un

rôle plus crucial que le montant pur de la transaction. Cela peut signifier que la fraude n'est pas forcément associée à de grosses transactions, mais à certains types de commerçants ou catégories.

4. heure_jour (0.028031)

- Interprétation : L'heure de la transaction dans la journée a une certaine influence, mais elle est moins marquée que d'autres caractéristiques.

- Explication pour les équipes métiers : Les fraudes semblent être un peu plus susceptibles de se produire à certaines heures de la journée, mais ce n'est pas un facteur déterminant. Cependant, cela pourrait suggérer que certains comportements de fraude surviennent plus souvent à des moments particuliers, comme la nuit ou pendant les périodes de faible surveillance.

5. Categorie_marchand_grocery_net (0.025371)

- Interprétation : Les transactions en ligne dans la catégorie épicerie ont une certaine influence, mais elle est faible comparée à celle des transactions en magasin (POS).

- Explication pour les équipes métiers : Cela indique que les transactions en ligne dans le secteur de l'épicerie peuvent également être un indicateur de fraude, mais peut-être à un moindre degré par rapport aux transactions physiques. Cela pourrait aussi suggérer un changement dans les comportements des fraudeurs qui se tournent de plus en plus vers le commerce en ligne.

6. Categorie_marchand_home (0.021430)

- Interprétation : Les transactions liées à la catégorie maison montrent une faible mais significative influence.

- Explication pour les équipes métiers : Les achats dans la catégorie maison peuvent aussi être un domaine de fraude, comme des achats en ligne non autorisés ou de fausses déclarations concernant les achats dans des magasins de cette catégorie.

Autres catégories (ex. Categorie_marchand_travel, Categorie_marchand_misc_net, etc.)

- Interprétation : Les autres catégories de marchands (ex. voyages, divers en ligne et en magasin) ont une influence moins marquée sur le modèle.

- Explication pour les équipes métiers : Ces catégories représentent probablement des secteurs moins susceptibles d'être associés à des comportements frauduleux par rapport aux secteurs précédemment mentionnés, bien qu'il soit toujours possible que des fraudes se produisent dans ces domaines. Il pourrait être utile de surveiller ces secteurs, mais ils ne semblent pas aussi cruciaux que l'épicerie ou le carburant/transport.

Conseils pour les équipes métiers :

1. Surveiller les transactions dans les épiceries (magasins physiques) : Puisque la catégorie épicerie en magasin est l'une des plus influentes, il peut être utile d'ajuster les alertes pour les transactions de ce type.
2. Cibler les transactions dans les stations-service et transport : Étant donné que la catégorie carburant/transport joue également un rôle important, une surveillance renforcée pourrait être mise en place sur ces types de transactions.

3. Analyser les transactions en ligne dans certaines catégories : Bien que moins influentes, les transactions en ligne dans l'épicerie et la maison montrent une petite influence, ce qui pourrait indiquer que des fraudes se produisent dans ces catégories également.

En résumé, il semble que les fraudes soient plus fréquentes dans certains types de commerçants, comme les épiceries et stations-service, et que des comportements frauduleux peuvent aussi être observés dans des transactions en ligne, bien que dans une moindre mesure.

Si tu veux des explications plus détaillées sur certains aspects ou des visuels spécifiques pour l'équipe métier, n'hésite pas à me le dire !

Dans le contexte de la **détection de fraude**, on fait généralement référence à des comportements suspects ou malveillants liés à des **transactions financières**. Ces transactions peuvent être frauduleuses pour différentes raisons, et le terme "fraude" peut recouvrir plusieurs types de comportements. Voici quelques exemples spécifiques à ton domaine :

1. Utilisation de cartes de paiement volées ou compromises :

- **Exemple** : Un fraudeur utilise une carte bancaire volée pour effectuer une transaction en ligne ou en magasin. La fraude peut aussi être liée à un vol d'identifiants de paiement.

- **Impact** : La personne titulaire de la carte peut ne pas être au courant de l'utilisation de sa carte pour des achats frauduleux, et la transaction est généralement annulée une fois le vol détecté.

2. Transactions non autorisées par le titulaire du compte :

- **Exemple** : Un titulaire de carte bancaire n'a pas autorisé la transaction, mais un fraudeur a pu effectuer un achat en ligne en utilisant ses informations bancaires, ou une transaction en magasin a eu lieu sans son consentement.
- **Impact** : Ce type de fraude concerne les cas où une personne utilise les informations de paiement d'un autre sans autorisation.

3. Fabrication de fausses transactions (chargeback fraud) :

- **Exemple** : Un fraudeur effectue un achat en ligne, puis contacte l'émetteur de la carte pour signaler que la transaction était frauduleuse (chargeback), bien que ce soit un achat légitime qu'il a effectué.
- **Impact** : Cette forme de fraude est particulièrement difficile à détecter, car la transaction initiale est légitime, mais le fraudeur abuse des procédures de contestation.

4. Transactions suspectes ou anormales :

- **Exemple** : Un utilisateur effectue une série de transactions dans des magasins ou des catégories de marchands inhabituels pour lui. Par exemple, un achat important dans une **épicerie** alors que le client ne fait habituellement pas de gros achats dans cette catégorie.
- **Impact** : Ces types de comportements peuvent signaler que le titulaire de la carte a été victime de fraude (par exemple, un achat effectué sans son consentement), ou que la personne utilise de fausses informations de paiement.

5. Fraude interne ou manipulation des systèmes :

- **Exemple** : Un employé d'une institution bancaire ou d'une société de paiement peut manipuler les systèmes pour effectuer des transactions frauduleuses en utilisant les comptes de clients ou des informations de paiement internes.
- **Impact** : Ce type de fraude est souvent difficile à détecter, car les employés connaissent les systèmes de l'entreprise et peuvent les exploiter à des fins frauduleuses.

6. Fraude par money laundering (blanchiment d'argent) :

- **Exemple** : Des transactions répétées ou inhabituelles peuvent être un signe de blanchiment d'argent, où l'argent est déplacé de manière à dissimuler sa provenance illicite.
- **Impact** : Bien que les transactions puissent sembler légitimes, elles servent en réalité à nettoyer des fonds obtenus par des activités illégales.

7. Fraude par synthetic identity (identité synthétique) :

- **Exemple** : Une personne crée une fausse identité en combinant des informations réelles et fictives (par exemple, des numéros de sécurité sociale ou des dates de naissance volées) pour ouvrir des comptes bancaires ou faire des achats en ligne.
- **Impact** : Cela permet à l'individu de faire des achats frauduleux sans laisser de trace directe à une seule personne réelle.

Comment la détection de fraude fonctionne :

Le **modèle de détection de fraude** (comme celui que tu utilises avec XGBoost) est souvent conçu pour repérer ces types de comportements en se basant sur des **modèles de machine learning** qui identifient des **anomalies** ou des **patterns** inhabituels dans les transactions. Voici les grandes lignes de ce processus :

- **Données historiques** : Le modèle est formé à partir de données de transactions passées, comprenant à la fois des transactions légitimes et des transactions frauduleuses.
- **Caractéristiques comportementales** : Le modèle analyse des caractéristiques comme les **catégories de marchands**, les **montants des transactions**, l'**heure de la journée**, et le **comportement d'achat** global d'un utilisateur.
- **Comparaison avec des comportements attendus** : Il compare les nouvelles transactions à des modèles d'achats "normaux". Si une transaction présente des caractéristiques similaires à celles d'une fraude passée (par exemple, des achats inhabituels à certaines heures ou dans des catégories de produits spécifiques), elle est marquée comme potentiellement frauduleuse.

Pour résumer :

Une **fraude** dans ton contexte fait référence à une **transaction illégitime ou non autorisée** effectuée avec de **fausses informations** ou dans des circonstances suspectes. Le modèle que tu as formé aide à identifier ces transactions à partir des **données des utilisateurs**, en se basant sur des **patterns** connus de fraude et des comportements **anormaux** dans les transactions.

Est-ce que tu souhaites approfondir un de ces types de fraude, ou tu aimerais des exemples plus spécifiques de détection dans un certain contexte ?

Mise en production :

Pour la mise en production de mon projet, j'ai déjà développé une API Flask permettant le déploiement de mon modèle en environnement de test. Cette étape a permis de valider le fonctionnement du modèle et de l'exposer via une interface simple.

Actuellement, je suis en train de me former sur Google Cloud Platform (GCP) afin d'avancer sur la mise en production du modèle à plus grande échelle, en intégrant des outils comme Docker pour assurer un déploiement efficace et scalable.