Note Méthodologique

1. Méthodologie d'Entraînement du Modèle

La démarche adoptée pour l'entraînement du modèle s'est effectuée en plusieurs étapes cruciales visant à garantir l'efficacité et la précision du modèle final. Voici un résumé détaillé des étapes suivies :

a. Préparation des Données :

Avant l'entraînement, les données ont été séparées en trois ensembles distincts : entraînement, validation, et test. Cette segmentation permet d'entraîner le modèle sur un ensemble de données, de valider les performances sur un autre, et de tester la robustesse du modèle sur un ensemble de données inédit.

b. Traitement des Variables :

Le traitement des variables catégorielles et continues a été réalisé pour assurer un format approprié pour l'entraînement du modèle. Cela inclut la gestion des valeurs manquantes, l'encodage des variables catégorielles, et la normalisation des variables continues.

c. Features Selections :

Une étape cruciale qui vise à réduire la dimensionnalité des données et à se concentrer sur les caractéristiques les plus pertinentes pour la tâche de prédiction.

d. Optimisation des Hyperparamètres :

Un processus d'optimisation bayésienne a été utilisée pour optimiser les hyperparamètres du modèle LightGBM, en utilisant la fonction coût métier comme métrique d'optimisation.

e. Entraînement du Modèle :

Avec les hyperparamètres optimisés, le modèle LightGBM a été entraîné sur l'ensemble d'entraînement.

f. Évaluation et Validation :

Enfin, le modèle a été évalué sur l'ensemble de validation en utilisant diverses métriques pour s'assurer de sa robustesse et de sa performance.

2. Traitement du Déséquilibre des Classes

Le jeu de données initial a montré un déséquilibre significatif entre les classes. Pour remédier à cela, deux approches ont été adoptées :

a. Over-sampling avec SMOTE :

La technique SMOTE a été utilisée pour générer des données synthétiques pour la classe minoritaire, enrichissant ainsi l'ensemble de données avec des informations supplémentaires.

b. Under-sampling :

Cette technique a réduit la classe majoritaire pour équilibrer la distribution des classes, en veillant à éviter le surajustement vers la classe majoritaire.
Ces approches ont été intégrées dans la pipeline d'entraînement pour garantir un apprentissage équilibré et efficace du modèle.

3. Fonction Coût Métier, Algorithme d'Optimisation et Métrique d'Évaluation

a. Fonction Coût Métier :

Une fonction coût métier personnalisée a été conçue pour mieux refléter l'impact métier des erreurs de prédiction.

b. Algorithme d'Optimisation :

L'algorithme de boosting intégré dans LightGBM a été utilisé pour minimiser la fonction de coût.

c. Métrique d'Évaluation :

Le modèle a été évalué en utilisant la métrique ROC-AUC, qui donne une indication de la performance du modèle indépendamment du seuil de classification choisi.

4. Tableau de Synthèse des Résultats

Voici un tableau résumant les valeurs des métriques considérées :

Métrique	Valeur
ROC-AUC	0.67
Coût Métier	0.68

5. Interprétabilité Globale et Locale du Modèle

L'interprétabilité du modèle a été analysée en utilisant SHAP (SHapley Additive exPlanations).

a. Interprétabilité Globale :

Les valeurs SHAP ont été utilisées pour évaluer l'importance globale des caractéristiques dans le modèle.

b. Interprétabilité Locale :

Des visualisations SHAP force plot ont été générées pour certains échantillons individuels, permettant une compréhension claire de l'influence des caractéristiques sur les prédictions du modèle au niveau local.

6. Limites et Améliorations Possibles

a. Limites :

Sensibilité aux outliers, bruit dans les données et peut-être surajustement dû au déséquilibre des classes.

b. Améliorations :

Essayer d'autres techniques de rééquilibrage des classes, explorer des architectures de modèles alternatifs, et peut-être intégrer des techniques d'ensemble pour améliorer la robustesse du modèle. Une autre piste pourrait être l'expérimentation d'une ingénierie des caractéristiques plus avancée pour extraire davantage d'informations pertinentes des données.

7. Analyse du Data Drift

L'analyse du Data Drift a été produit à l'aide d'Evidently. On a mesuré la présence de Data Drift entre le dataset train et le dataset test à l'aide du fichier HTML.

a. Type :

Donnant le type de colonne (int, str, obj).

b. Reference vs Current Distribution :

Met en relation les distributions du data set de train avec celui de set pour voir d'éventuel changement (et donc d'identifier du Data Drift)

c. Data Drift :

Colonne évaluant la présence de data drift ou non. Uniquement 9 features font du data drift sur 121, cela est bien trop peu pour considérer qu'il y a globalement du Data drift.

d. Drift Score :

Score qui lorsqu'il dépasse le seuil de 0.1 prévient d'un potentiel Data Drift.