# Méthodologie d’entraînement du modèle

## Objectif de la modélisation

L’objectif de ce projet est de développer un modèle de scoring crédit capable d’estimer la probabilité de défaut d’un client à partir de données socio-démographiques et financières, puis de transformer cette probabilité en une décision opérationnelle d’octroi ou de refus de crédit. Le problème est formulé comme une **classification binaire supervisée**, dans un contexte de **déséquilibre important des classes**, la classe des clients défaillants étant minoritaire.La méthodologie d’entraînement vise à garantir des performances robustes, une évaluation réaliste et une optimisation alignée avec les enjeux métier.

## Préparation des données et découpage

Plusieurs stratégies d’imputation des valeurs manquantes ont été testées (par exemple *median/mode*, *médian/missing*) afin d’évaluer la robustesse des modèles face à la qualité des données. Chaque version du dataset a été considérée comme une configuration distincte lors de la phase de comparaison.

Le jeu de données d’entraînement a été découpé en sous-ensembles **train**, **validation** et **test** selon une stratégie stratifiée, afin de préserver la distribution des classes dans chaque sous-ensemble.Ce découpage permet d’éviter toute fuite d’information et d’obtenir une estimation fiable des performances en conditions réelles.

## Pipelines et prétraitement

Tous les modèles ont été intégrés dans des **pipelines scikit-learn** afin d’assurer la cohérence et la reproductibilité des traitements.

Les variables catégorielles ont été encodées par **One-Hot Encoding** avec gestion des modalités rares et des valeurs inconnues. Les variables numériques ont été transmises sans mise à l’échelle pour les modèles basés sur des arbres étant peu sensibles à la normalisation.Cette approche garantit que les mêmes transformations sont appliquées de manière identique lors de l’entraînement et de l’évaluation.

## Comparaison initiale des modèles

Une première phase de comparaison a été menée entre plusieurs familles de modèles avec un modèle naïf de référence(*Dummy Classifier*), les modèles utilisés pour la baseline (régression logistique et histogram gradient boosting), XGBoost et LightGBM.

Des hyperparamètres simples ont été utilisés afin de comparer les algorithmes sur des bases équitables. Les modèles ont été évalués à l’aide de métriques classiques (AUC, rappel, précision et F1-score) ainsi que d’un **score métier** tenant compte du coût

asymétrique des erreurs.Cette phase a permis d'identifier **XGBoost** et **LightGBM** comme les modèles les plus prometteurs.

## Optimisation des hyperparamètres

Les modèles XGBoost et LightGBM ont ensuite été optimisés à l'aide d'**Optuna**, selon une approche itérative en plusieurs phases : une exploration large de l'espace des hyperparamètres, une recherche affinée autour des configurations les plus performantes et une analyse de la convergence avec un nombre élevé d'arbres.

Chaque essai a été évalué via une validation croisée stratifiée, et l'optimisation a été guidée par le **score métier moyen**, afin d'aligner directement la recherche d'hyper paramètres avec l'objectif final du projet.

## Sélection du modèle final

Les performances finales de XGBoost et LightGBM se sont révélées proches en termes de métriques et de coût métier. Le modèle **Light GBM** a été retenu comme modèle final en raison de performances comparables voire légèrement supérieures, un temps d'entraînement et d'inférence nettement plus faibles, et une meilleure simplicité d'intégration dans un contexte industriel.

## Optimisation du seuil de décision

Le seuil de décision transformant la probabilité prédite en classe binaire a été optimisé sur le jeu de validation en minimisant explicitement le **coût métier**. Ce seuil a ensuite été appliqué tel quel au jeu de test pour l'évaluation finale, afin de préserver l'indépendance des jeux de données.Cette démarche permet de dissocier l'apprentissage du modèle de la calibration décisionnelle, tout en évitant toute fuite d'information.

## Suivi expérimental

L'ensemble des expérimentations a été suivi à l'aide de **MLflow**, permettant de tracer les paramètres, métriques, temps de calcul et modèles entraînés, et garantissant la reproductibilité complète du processus d'entraînement.

# Traitement du déséquilibre des classes

## Contexte du déséquilibre

Le jeu de données présente un **déséquilibre marqué entre les classes**, la classe des clients défaillants étant nettement minoritaire. Dans un contexte de scoring crédit, ce déséquilibre est critique car une mauvaise prise en compte peut conduire à des modèles biaisés favorisant excessivement la classe majoritaire, au détriment de la détection des clients à risque.

## Approches évaluées

Deux stratégies ont été testées pour traiter ce déséquilibre :

- **Ajustement des poids de classes**
  Les modèles linéaires et les modèles de type gradient boosting ont été entraînés avec des pondérations inverses à la fréquence des classes (`class_weight="balanced"` ou `scale_pos_weight`). Cette approche permet de pénaliser davantage les erreurs sur la classe minoritaire sans modifier la distribution des données.

- **Undersampling**
  Une méthode de **sous-échantillonnage** aléatoire de la classe majoritaire qui va tout simplement *réduire* le nombre d'exemples de la *classe majoritaire* pour obtenir un *ratio plus équilibré*.

Les techniques de sur-échantillonnage ou de sous-échantillonnage (SMOTE, undersampling) n'ont pas été retenues afin de préserver la distribution réelle des données et d'éviter l'introduction de biais artificiels.

## Choix retenu

L'approche par **pondération des classes** a été retenue pour l'ensemble des modèles finaux, car elle s'intègre naturellement aux algorithmes utilisés, préserve la structure des données, améliore significativement le rappel sur la classe défaillante, reste cohérente avec un contexte industriel réel.

# Fonction coût métier, algorithme d'optimisation et métriques d'évaluation

## Définition de la fonction coût métier

Dans un contexte de crédit, les erreurs n'ont pas le même impact :

- **Faux négatif (FN)** : un client défaillant est accepté → coût élevé,
- **Faux positif (FP)** : un client solvable est refusé → coût plus faible.

Une fonction coût métier asymétrique a donc été définie avec le coût FP = 1 et le coût FN = 10. Le coût total est calculé comme :

Coût = **FP** X **1**+ **FN** X **10**

Cette fonction permet d'aligner l'optimisation du modèle avec les enjeux économiques réels.

## Algorithme d'optimisation

Les hyperparamètres des modèles XGBoost et LightGBM ont été optimisés à l'aide d'**Optuna**, en minimisant le **coût métier moyen** sur validation croisée stratifiée.

L'optimisation a été réalisée en plusieurs phases: une exploration large, une recherche affinée, et une validation avec un nombre élevé d'arbres (n_estimators).

Le seuil de décision a ensuite été optimisé indépendamment, en recherchant la valeur minimisant le coût métier sur le jeu de validation.

## Métriques suivies

En complément du coût métier, les métriques suivantes ont été analysées l'AUC ROC (capacité de discrimination), précision (postifs prédits parmis les toutes les prédictions positives), rappel (capacité du modèle à ne pas rater les positifs), F1-score et F1 macro pour évaluer l'équilibre entre classes.

Le coût métier reste cependant la métrique de décision principale.

# Tableau de synthèse des résultats

Le tableau ci-dessous présente les performances finales des deux modèles retenus sur le jeu de test :

| Modèle | AUC | Recall (défaillant) | F1-score | Coût métier | Temps d'entraînement |
|---|---|---|---|---|---|
| XGBoost | ≈ 0.79 | ≈ 0.69 | ≈ 0.30 | ≈ 29 900 | Élevé |
| LightGBM | ≈ 0.79 | ≈ 0.67 | ≈ 0.31 | ≈ 29 850 | Faible |

Les performances des deux modèles sont très proches en termes de qualité prédictive et de coût métier. XGBoost est meilleur sur le Recall mais LightGBM présente cependant un **avantage significatif en temps de calcul**, ce qui est déterminant dans une perspective industrielle.

De plus, notre métrique principale, le coût métier global, sur lequel on a optimisé les hyper paramètres de nos deux modèles est légèrement inférieur pour LGBM, c'est pourquoi on a décidé de garder LGBM pour la mise en production sur ce projet.

# Interprétabilité globale et locale du modèle

## Importance globale des variables

L'Interprétabilité du modèle final (Light GBM) a été analysée à l'aide de **SHAP (SHapley Additive exPlanations)**.
Les graphiques d'importance globale montrent que :

- les variables liées aux sources externes de score (ex. *EXT_SOURCE*) jouent un rôle central,
- certaines variables sociodémographiques ont un impact plus modéré,
- Les effets sont globalement cohérents avec l'intuition métier.

## Interprétabilité locale

Des analyses locales ont été réalisées pour des clients individuels :

- un client correctement classé comme solvable,
- un client correctement classé comme défaillant.

Les *waterfall plots* SHAP permettent de visualiser la contribution positive ou négative de chaque variable à la prédiction finale, rendant la décision du modèle explicable au niveau individuel.

## Apport métier

Cette interprétabilité :

- renforce la confiance dans le modèle,
- facilite la communication avec des parties non techniques,
- constitue un prérequis essentiel dans un contexte réglementé comme le crédit.

# Limites et améliorations possibles

## Limites du modèle

Les performances du modèle dépendent fortement des choix de préparation des données, en particulier des stratégies d'imputation (médiane, mode, valeurs manquantes explicites). Bien que plusieurs méthodes aient été comparées, ces approches restent relativement simples et peuvent ne pas capturer toute l'information contenue dans les valeurs manquantes.

L'optimisation s'est principalement concentrée sur les hyperparamètres des modèles et sur le seuil de décision basé sur la fonction coût métier. D'autres leviers importants, tels que la création de nouvelles variables métier ou une meilleure optimisation des hyperparamètres est envisageable pour une version ultérieure.

L'analyse d'interprétabilité via SHAP montre une cohérence globale satisfaisante, mais certaines variables présentent des relations plus diffuses, probablement dues à des interactions complexes ou à des corrélations entre variables. Cela limite la capacité à produire des règles métier simples et totalement explicables.

## Pistes d'amélioration

Plusieurs axes pourraient permettre d'améliorer les performances et la robustesse du modèle :

- enrichissement des features (interactions, statistiques groupées, encodages catégoriels avancés),
- utilisation de méthodes d'imputation plus sophistiquées,
- optimisation plus poussée orientée coût métier,
- analyse spécifique des erreurs (faux positifs et faux négatifs) à l'aide de SHAP.

# Analyse du Data Drift

## Objectif

Cette analyse vise à détecter une éventuelle dérive des données entre la phase d'entraînement du modèle de scoring et une situation de production simulée, afin d'anticiper une dégradation potentielle des performances du modèle dans le temps.

## Méthodologie

L'analyse du Data Drift est réalisée à l'aide de la librairie **Evidently**. Le jeu de données `application_train` est utilisé comme référence (données d'entraînement), tandis que `application_test` représente les données simulant la production. Evidently compare les distributions des variables entre les deux jeux de données en utilisant : la **distance de Wasserstein** pour les variables numériques et la **distance de Jensen-Shannon** pour les variables catégorielles.

Une variable est considérée en dérive si la distance dépasse un seuil statistique. Un Data Drift global est détecté uniquement si plus de **50 % des variables** sont concernées.

## Résultats globaux

Sur les **120 variables** analysées, **9 présentent un Data Drift**, soit **7,5 %** des variables. Le seuil global n'étant pas atteint, **aucun Data Drift critique n'est détecté à l'échelle du dataset**. Les variables en dérive concernent principalement : les **montants financiers** (`AMT_CREDIT`, `AMT_GOODS_PRICE`, `AMT_ANNUITY`), les **consultations du bureau de crédit**, certaines **variables catégorielles** (`FLAG_EMAIL`, `NAME_CONTRACT_TYPE`). À l'inverse, des variables importantes du modèle, telles que `EXT_SOURCE`, restent stables.

## Impact sur le modèle et recommandations

Bien que le dataset soit globalement stable, la dérive observée sur certaines variables clés peut, à terme, impacter la performance du modèle. Il est donc recommandé de mettre en place un **suivi régulier du Data Drift**, avec une attention particulière portée aux variables financières, et d'envisager un **réentraînement du modèle** en cas d'augmentation significative de la dérive..