NOTE METHODOLOGIQUE
OPEN CLASS ROOM – FORMATION DATASCIENTIST
PROJET 7 – IMPLEMENTER UN MODELE DE SCORING

Anaïs CHAPON_Octobre 2024

1. Méthodologie d'entraînement du modèle

1.1 Modèles testés

Dans ce projet, trois modèles ont été testés pour prédire nos données :

- **Dummy Regressor** : Le Dummy Regressor est un modèle de régression simple utilisé principalement comme **référence de base** pour évaluer les performances de modèles de régression plus complexes. Il ne cherche pas à capturer les relations entre les variables d'entrée et la cible, mais prédit des valeurs basées sur des règles simples, comme la moyenne, la médiane, ou une valeur constante. Par exemple, un Dummy Regressor configuré pour utiliser la moyenne prédira cette même valeur pour toutes les observations. L'objectif principal de ce modèle est de fournir un point de comparaison minimal afin de vérifier si un modèle plus sophistiqué apporte une réelle amélioration par rapport à une stratégie de prédiction triviale. Différentes stratégies de Dummy Regressor ont été testé : 'stratified', 'most_frequent', 'prior', 'uniform', 'constant'.

- **Regression logistique** : La régression logistique est une méthode statistique utilisée pour modéliser la relation entre une ou plusieurs variables indépendantes et une variable dépendante binaire (c'est-à-dire avec deux classes possibles, comme 0 et 1). Contrairement à la régression linéaire, qui prévoit une valeur continue, la régression logistique utilise la fonction sigmoïde pour produire une probabilité comprise entre 0 et 1, indiquant la probabilité que l'événement étudié se produise. Lorsque cette probabilité dépasse un certain seuil (généralement 0,5), la classe positive est prédite. Ce modèle est largement utilisé dans les domaines de la classification, tels que le diagnostic médical, la détection de fraudes et les systèmes de crédit, en raison de sa simplicité et de son interprétabilité.

- **Light GBM** : LightGBM (Light Gradient Boosting Machine) est un puissant algorithme de machine learning basé sur les **arbres de décision**, développé par Microsoft, qui utilise une approche de gradient boosting pour les tâches de classification et de régression. Il est conçu pour être très rapide et efficace, avec une capacité à gérer de grands ensembles de données tout en maintenant une haute performance. LightGBM se distingue par sa manière unique de construire les arbres de décision, utilisant une technique appelée leaf-wise growth qui optimise les feuilles les plus prometteuses, ce qui permet une réduction plus rapide de l'erreur par rapport aux approches traditionnelles basées sur la profondeur. En plus d'être plus rapide, LightGBM consomme moins de mémoire et offre une meilleure précision, ce qui le rend particulièrement adapté aux projets nécessitant une grande évolutivité et des temps de calcul réduits.

1.2 Prétraitement

Pour le Dummy Regressor et la régression logistique, il est nécessaire de faire une imputation sur les données : il s'agit d'une technique utilisée pour **gérer les valeurs manquantes** dans un jeu de données. Lorsqu'une donnée est absente pour une variable numérique, cette méthode consiste à remplacer la valeur manquante par une autre valeur : nous avons choisi ici de la remplacer par la **moyenne** des valeurs non manquantes de cette même variable.

1.3 Standardisation des données

Le Standard Scaler est une technique de **standardisation des données** utilisée pour transformer les caractéristiques d'un jeu de données afin qu'elles aient une moyenne de 0 et un écart-type de 1. Cette mise à l'échelle est importante pour de nombreux algorithmes de machine learning, comme ceux basés sur les distances (par exemple, la régression logistique ou les réseaux de neurones), car elle garantit que chaque caractéristique contribue de manière égale au modèle, indépendamment de son amplitude initiale. Le Standard Scaler fonctionne en soustrayant la moyenne de chaque caractéristique et en divisant le résultat par son écart-type, ce qui permet d'améliorer la convergence des algorithmes d'optimisation et de stabiliser leurs performances.

1.4 Validation croisée

La validation croisée est une technique essentielle en machine learning utilisée pour **évaluer la performance d'un modèle** de manière robuste. Elle permet de s'assurer que les résultats obtenus par le modèle sont généralisables et qu'ils ne sont pas simplement dus à un surapprentissage (overfitting) sur le jeu de données d'entraînement. Il existe différentes méthodes de validation croisée, nous avons choisi ici **StratifiedKFold** avec 5 folds.

1.5 Hyperparamètre seuil

Les modèles de classification produisent une probabilité en sortie, permettant de déterminer si l'octroi d'un crédit à un client peut être accordé ou non. Habituellement, la valeur par défaut du seuil de décision est fixée à 0,5. Toutefois, ce seuil peut ne pas toujours être optimal pour toutes les situations. En ajustant ce seuil de manière appropriée, il devient possible de mieux équilibrer et d'optimiser le nombre de faux positifs et de faux négatifs, améliorant ainsi la précision du modèle selon les objectifs spécifiques. Ici, le seuil final est établi à **0,52**.

1.6 MLflow

MLflow est une plateforme open-source conçue pour **gérer le cycle de vie des modèles** de machine learning. Il offre une suite d'outils permettant de suivre les expérimentations, de gérer les modèles, et de faciliter leur déploiement.

2. Traitement du déséquilibre des classes

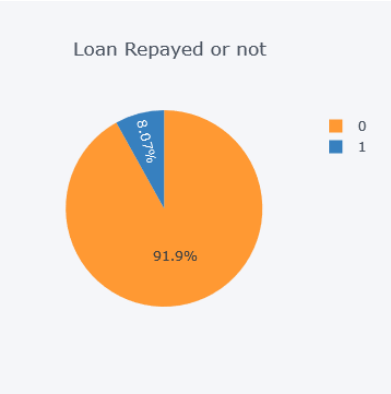

Fig 1 : Répartition de la variable TARGET

Le déséquilibre des classes est un problème courant en machine learning qui survient lorsque les classes dans le jeu de données sont représentées de manière inégale, c'est-à-dire qu'une classe est beaucoup plus fréquente que les autres. Ce déséquilibre peut fortement affecter la performance des modèles de classification, car ceux-ci tendent à favoriser la classe majoritaire au détriment des classes minoritaires. Par exemple, dans le cas d'un modèle de détection de fraudes, s'il y a beaucoup plus de transactions légitimes que frauduleuses, le modèle pourrait simplement prédire que toutes les transactions sont légitimes pour obtenir une précision élevée, tout en ignorant les fraudes réelles. Cette situation entraîne un taux élevé de faux négatifs, ce qui peut être coûteux dans des applications critiques. Pour pallier ce problème, des techniques comme le rééchantillonnage des données, l'ajustement des seuils de décision ou l'utilisation de métriques adaptées comme le F1-score sont souvent utilisées pour garantir que le modèle tient compte des classes minoritaires.

Pour gérer le problème des classes déséquilibrées en machine learning, deux approches courantes sont la méthode **SMOTE** (Synthetic Minority Over-sampling Technique) et l'ajustement des **class weights** (poids de classes). Ces techniques permettent de mieux équilibrer la contribution des classes minoritaires dans le processus d'apprentissage.

SMOTE est une méthode de rééchantillonnage qui crée artificiellement de nouvelles instances de la classe minoritaire en générant des exemples synthétiques plutôt que de simplement dupliquer les données existantes. L'objectif de SMOTE est de donner plus de poids à la classe minoritaire, permettant ainsi au modèle d'apprendre des motifs pertinents dans cette classe, tout en réduisant le risque de surapprentissage (overfitting) qui pourrait survenir avec une simple duplication.

L'ajustement des **class weights** consiste, quant à lui, à modifier les poids attribués à chaque classe dans la fonction de perte utilisée par le modèle. Dans cette approche, on accorde un poids plus élevé aux erreurs commises sur la classe minoritaire et un poids plus faible sur la classe majoritaire. Cela incite le modèle à accorder plus d'importance à la prédiction correcte des exemples de la classe minoritaire.

Les deux méthodes ont été testées pour nos modèles, et la méthode choisie finalement sera **Class Weights.**

3. Fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation

3.1 Fonction coût métier

La fonction coût métier est un concept clé en machine learning et en statistique, utilisé pour quantifier les conséquences financières ou opérationnelles des erreurs de classification commises par un modèle.
Dans notre cas, il existe deux types d'erreur de prédiction :

- **Les Faux Positifs (FP)** : Il s'agit de situations où la banque juge qu'un client ne pourra pas rembourser son prêt, alors qu'en réalité, il en est capable.
- **Les Faux Négatifs (FN)** : Ce sont des cas où la banque pense que le client sera en mesure de rembourser le crédit, mais celui-ci finit par faire défaut.

Ces deux erreurs ne présentent pas les mêmes conséquences pour la banque : un faux négatif sera bien plus coûteux. On va donc créer un indicateur, le coût métier, qui propose un poids plus lourd pour les Faux Négatifs que les Faux Positifs :

Coût métier = Somme (10*FN + FP)

3.2 Algorithme d'optimisation

Un algorithme d'optimisation est une méthode ou un processus utilisé pour trouver la meilleure solution à un problème donné, en **minimisant ou en maximisant une fonction objective**. En d'autres termes, il s'agit d'une technique qui cherche à ajuster des paramètres pour obtenir les résultats les plus performants possibles, en fonction de certains critères ou contraintes.
Dans ce projet, on utilise l'algorithme **Optuna**.

3.3 Métriques d'évaluation

Nous utilisons différentes métriques pour évaluer nos modèles :

AUC (Area Under the Curve) : Mesure la capacité du modèle à classer correctement les classes positives et négatives. Plus la valeur de l'AUC est élevée (proche de 1), meilleure est la performance du modèle.

Accuracy (Précision) : C'est le rapport entre le nombre de prédictions correctes et le nombre total de prédictions. Bien qu'elle soit simple à comprendre, l'accuracy peut être trompeuse dans le cas de classes déséquilibrées.

Business Score : Cette métrique est cruciale pour évaluer l'impact économique des erreurs de classification dans le contexte de votre application.

Fit Time (Temps d'entraînement) : Mesure le temps nécessaire pour entraîner le modèle.

Prediction Time (Temps de prédiction) : Mesure le temps nécessaire pour faire des prédictions après l'entraînement du modèle.

4. Un tableau de synthèse des résultats

Modèle	Strategy	AUC	Accuracy	Business Score	Fit Time	Prediction Time
Dummy Regressor	Most_frequent	0.5	0.919	248030	0.009	2.27
	prior	0.5	0.919	248030	0.008	2.22
	uniform	0.5	0.919	248030	0.012	2.19
	stratified	0.5	0.851	251515	0.016	2.53
	constant	0.5	0.081	282425	0.000	2.17
Regression logistique		0.74	0.716	169350	0.02	2.39
Light GBM		0.73	0.704	172450	159.08	0.00

Nous choisirons donc les paramètres suivants :

- Modèle : Light GBM
- Paramètres : 'lr': 0.009047806551462997
  'num_leaves': 35
  'n_estimators': 255
  'threshold': 0.52

5 . L'interprétabilité globale et locale du modèle

5.1 L'interprétabilité globale

La **feature importance globale** mesure l'**impact de chaque caractéristique (feature) sur les prédictions d'un modèle** à l'échelle de l'ensemble des données. Elle permet d'identifier quelles variables ont le plus d'influence sur le comportement général du modèle, en fournissant une vue d'ensemble de l'importance relative de chaque caractéristique. Cette mesure est souvent utilisée pour sélectionner les variables les plus pertinentes et simplifier le modèle, tout en aidant à comprendre les relations sous-jacentes entre les variables d'entrée et les sorties. Les méthodes courantes pour évaluer la feature importance globale incluent l'analyse des gains de performance, des mesures comme la permutation importance, ou l'utilisation d'algorithmes intégrés tels que ceux dans les arbres décisionnels (par exemple, Gini importance ou réduction de l'impureté).

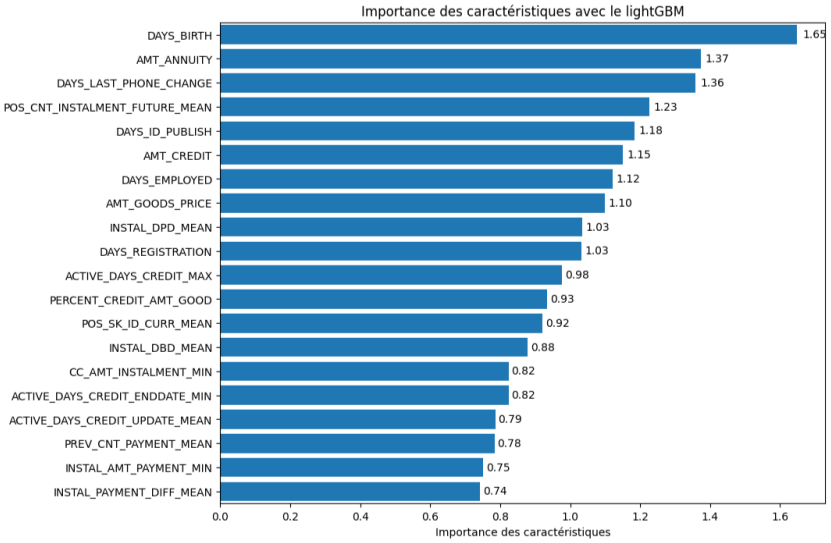

Fig 2 : Feature importance globale pour le Light GBM

5.2 Interprétabilité locale

La **feature importance locale**, en revanche, se concentre sur l'importance des caractéristiques pour des instances spécifiques ou des **prédictions individuelles**. Elle permet d'expliquer pourquoi un modèle a pris une certaine décision pour un cas particulier, en évaluant l'influence de chaque caractéristique sur la prédiction de cet exemple donné. Cela est particulièrement utile pour comprendre les décisions d'un modèle dans des contextes où des explications sont nécessaires, comme dans le domaine médical ou financier. Des techniques telles que **SHAP** (SHapley Additive exPlanations) ou **LIME** (Local Interpretable Model-agnostic Explanations) sont souvent utilisées pour générer des explications de feature importance locale, fournissant des insights sur comment chaque caractéristique a contribué à la décision finale pour chaque observation. Ici, nous avons utilisé **SHAP**.

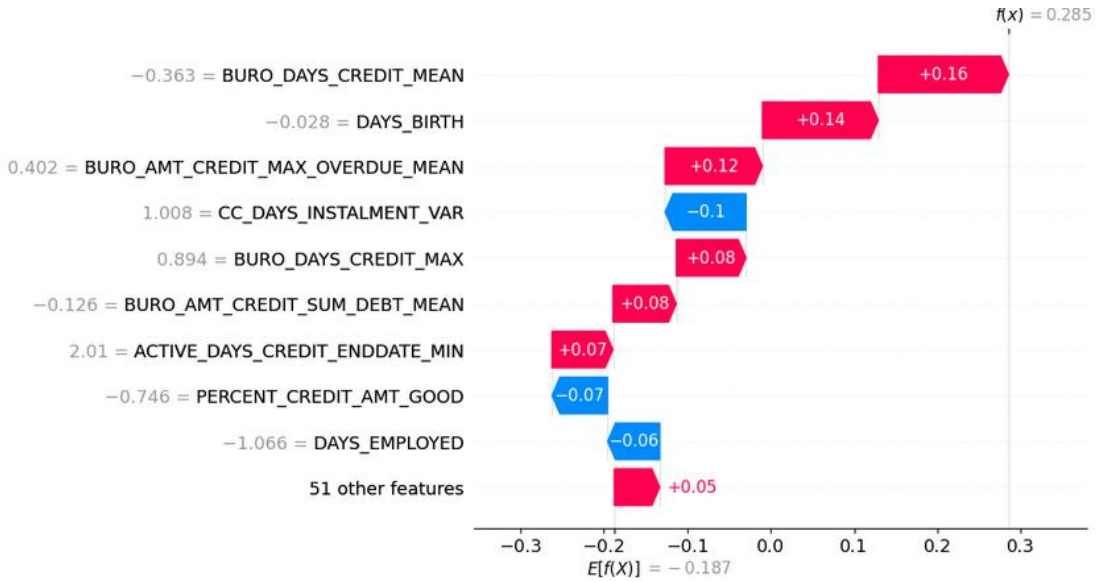

Fig 3 : Exemple de feature importance locale pour un individu

6. Les limites et les améliorations possibles

Des échanges avec des utilisateurs permettraient de mieux orienter la modélisation et la visualisation des données, ainsi que leur compréhension.
Le jeu de données étant vaste et complexe, une optimisation de la gestion des données aurait aussi pu aider à le rendre plus effectif.

7. L'analyse du Data Drift

Le data drift (ou dérive des données) désigne le phénomène par lequel la distribution des données d'entrée change au fil du temps par rapport aux données sur lesquelles un modèle de machine learning a été initialement entraîné. Ces changements peuvent affecter la performance du modèle, car il se base sur les relations qu'il a apprises à partir de la distribution des données d'entraînement, et toute variation dans ces données peut entraîner des prédictions moins précises ou même incorrectes.

L'hypothèse retenue est que le dataset "application_train" représente les données utilisées pour la modélisation, tandis que le dataset "application_test" correspond aux nouveaux clients lorsque le modèle est en production. Pour analyser les dérives des données, un test statistique de Kolmogorov-Smirnov sera appliqué aux colonnes numériques, et l'indice de stabilité de la population (PSI) sera utilisé pour les colonnes catégorielles. Un seuil de 0,2 a été défini pour chaque test statistique :

Si la p-value du test K-S est inférieure à 0,2, cela indique que les deux échantillons sont probablement issus de distributions différentes, avec un niveau de confiance de 80 %.

Si le PSI est supérieur à 0,2, cela suggère également une présence de Data Drift. Le package "evidently" sera utilisé pour mener cette analyse de Data Drift et pour générer une page HTML permettant d'examiner les résultats. Il est considéré que si plus de la moitié des variables montrent une dérive, alors un Data Drift global est détecté dans les données.

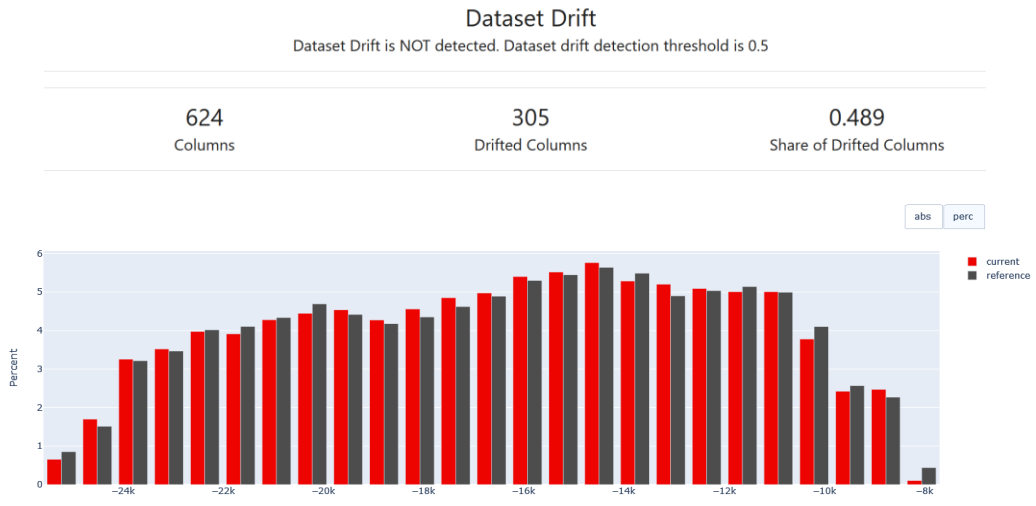

Fig 4 : Exemple de l'année de naissance