
- Table des Matières -

1.	Introduction.....	2
2.	Méthodologie d'entraînement	3
2.1	Données sources.....	3
2.2	Prétraitement	4
2.3	Processing.....	4
3.	Optimisation	5
3.1	Algorithme d'optimisation.....	5
3.2	Métrique d'évaluation	5
3.3	Fonction gain	5
4	Interprétabilité du modèle	7
5	Limites et améliorations possibles	8

1. Introduction

La présente **note méthodologique** a pour objectif de décrire :

- la méthodologie d'entraînement d'un modèle de classification,
- la fonction coût, l'algorithme d'optimisation et la métrique d'évaluation,
- l'interprétabilité du modèle et
- les limites et les améliorations possibles.

L'objectif du modèle mentionné tout au long de ce document est de calculer une prédiction sur la probabilité de faillite d'un client de façon automatique.

Les données utilisées à cette fin sont disponibles à l'adresse : <https://www.kaggle.com/c/home-credit-default-risk/data>

2. Méthodologie d'entraînement

2.1 Données sources

Le jeu de données est composé de 9 fichiers jeux de données :

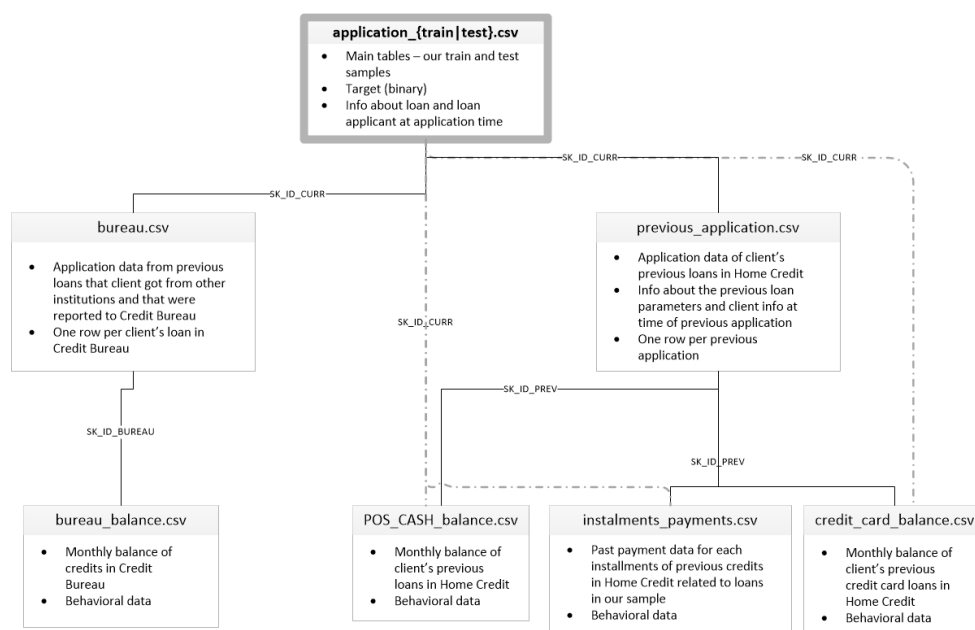
- 2 jeux principaux :
 - 'application_train.csv'
 - 'application_test.csv'
- 6 jeux secondaires :
 - 'bureau_balance.csv'
 - 'bureau.csv'
 - 'credit_card_balance.csv'
 - 'installments_payments.csv'
 - 'POS_CASH_balance.csv'
 - 'previous_application.csv'
- 1 fichier de description des variables des 8 précédents fichiers :
 - 'HomeCredit_columns_description.csv'

La classe que le modèle doit permettre de classer est la variable 'TARGET' (cible). La valeur '1' indique qu'un client a rencontré des difficultés de paiement (plus de X jours de retard de paiement sur les Y premières mensualités). La valeur '0' correspond à tous les autres cas.

L'analyse du fichier de descriptions des variables indique que l'ensemble du jeu représente :

- 219 variables agrégées
- 196 variables uniques,
- dont 14 présentes dans plusieurs tables

Le schéma suivant reprend les relations entre l'ensemble de ces tables.



Une pré-analyse met en avant que les classes sont déséquilibrées :

- classe 0 = 92% des prêts et
- classe 1 = 8% des prêts.

2.2 Prétraitement

Le prétraitement s'appuie sur une adaptation du kernel 'LightGBM with Simple Features' proposé par 'A.Assklou _ Aguiar'. La partie reprise du code consiste principalement en l'agrégation des tables et la création de variables statistiques (moyenne, minimum, maximum, décompte). Il supprime également certaines valeurs aberrantes.

Ce traitement est complété par :

- la suppression des variables n'ayant aucune valeur ('np.nan'),
- la suppression des valeurs infinies ('-np.inf' ou 'np.inf')
- l'imputation des valeurs manquante par la moyenne pour une variable partiellement renseignée.

2.3 Processing

Seules les données d'entraînement sont retenues (c'est-à-dire celles ayant une valeur renseignée pour 'TARGET'). Ces données filtrées sont dissociées en une cible 'y' (la variable 'TARGET') et une source X (toutes les autres variables).

La source est ensuite adaptée puis transformée en X_std avec le StandardScaler de scikit-learn.

Les jeux de test et d'entraînement sont définis avec le 'model_selection.train_test_split' de scikit-learn et les paramètres suivants :

- X = 'X_train_std',
- Y = 'y',
- test_size = 0.3 et
- stratify = 'y'

Ce dernier paramètre permet de corriger le biais dans la distribution des classes mentionné plus tôt.

3. Optimisation

3.1 Algorithme d'optimisation

Le modèle est optimisé par itérations sur la recherche des valeurs optimales de 2 de ces hyper paramètres avec GridSearchCV :

- 'max_depth':
- 'num_leaves'

Pour chacun de ces hyper paramètres, un premier passage avec le GridSearchCV est réalisé avec 5 valeurs étendues. [2, 5, 10, 20, 50, 100]

Le couple des valeurs optimales identifié sert de base a un second passage avec le GridSearchCV et 5 nouvelles valeurs sélectionnées avec la logique suivante :

- La valeur centrale est la valeur optimale,
- les 2 valeurs inférieures sont comprises entre la valeur optimale et la valeur la précédant dans le premier passage et
- les 2 valeurs supérieures sont comprises entre la valeur optimale et la valeur la suivante dans le premier passage.

3.2 Métrique d'évaluation

Lors de la phase d'optimisation, le modèle est évalué de manière technique sur la base du score AUC. Ceci permet d'estimer la proximité avec un classifieur parfait en traitant séparément la précision et le rappel, qui ont chacun des impacts financiers différents. Afin de déterminer le seuil de classification permettant de maximiser la marge de Home Credit, la fonction 'margins' est développée.

3.3 Fonction gain

Le tableau ci-dessous reprend les différentes typologies de clients

		Classe réelle	
		0	1
Classe prédite	0	TN (Vrai Négatifs)	FN (Faux Négatifs)
	1	FP (Faux Positifs)	TP (Vrais Positifs)

Chaque type de clients aura une contribution financière différente pour l'organisation :

- les clients en classe réelle 0 vont contribuer positivement aux résultats de l'organisation en amenant une marge,
- les clients en classe réelle 1 vont contribuer négativement aux résultats de l'organisation en amenant une perte,
- les clients en classe prédite 0 seront validés par le modèle et intégreront le portefeuille de clients historiques et
- les clients en classe prédite 1 ne seront pas validés par le modèle et ne contribueront donc pas aux gains ou perte de l'organisation.

Sur cette base, on peut en déduire une première formulation de la marge du modèle :

$$\text{Gain} = \text{Somme des marges des TN} - \text{somme des pertes des FN}$$

Afin de pouvoir estimer cette marge, la situation sera simplifiée en retenant les hypothèses suivantes :

- chaque client génère le même Chiffre d'Affaires (C.A.) moyen de 100 €
- la marge sur un dossier représente 10% du C.A. moyen et
- la perte sur un dossier = 100 % du C.A. moyen.

En intégrant ces hypothèses dans la formule, on obtient :

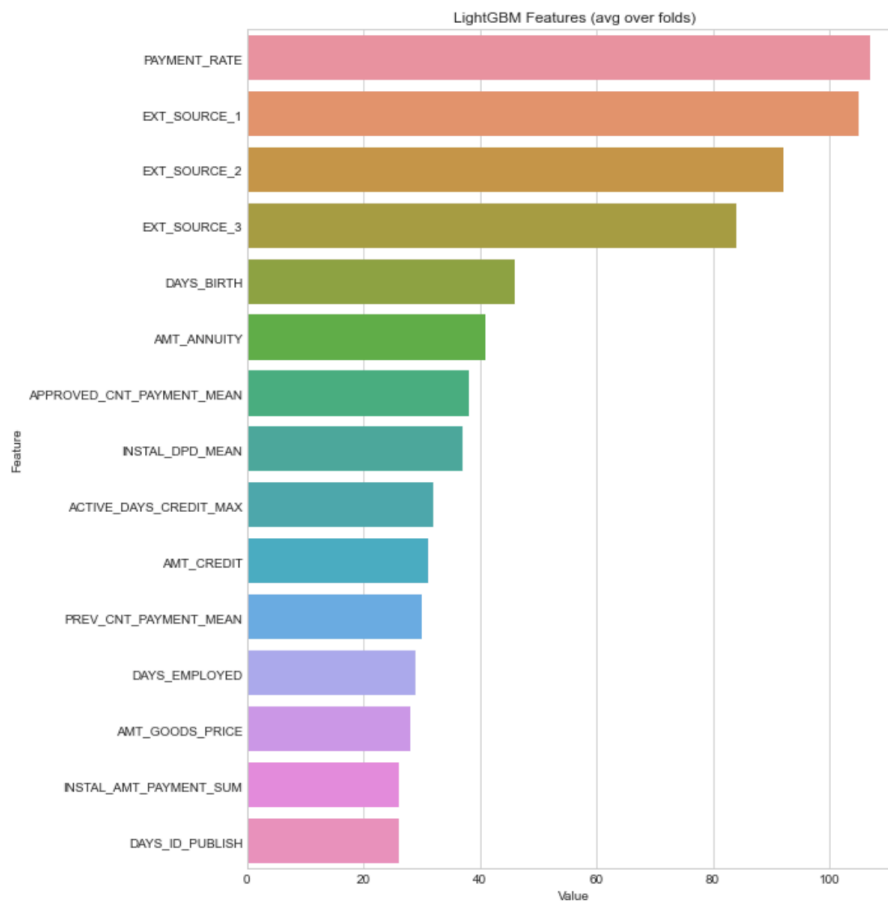
$$\begin{aligned}\text{Gain} &= [\text{TN} \times 10\% - \text{FN} \times 100\%] \times 100 \text{ €} \\ &= 10 \text{ €} \times \text{TN} - 100 \text{ €} \times \text{FN}\end{aligned}$$

La recherche du maximum de gain permet d'identifier le couple (TN, FN) donnant la classification optimale d'un point de vue métier.

4 Interprétabilité du modèle

Le classifieur LGBM possède dans sa bibliothèque la méthode 'feature_importances_'.

En appliquant cette méthode aux jeux de données traité, on obtient le poids de chaque variable dans le calcul du score de probabilité de classe. Le graphique ci-dessous indique par ordre décroissant d'importance les 15 premières variables les plus importantes :



Ce graphique met en avant que les variables :

- PAYMENT_RATE,
- EXT_SOURCE_1,
- EXT_SOURCE_2 et
- EXT_SOURCE_3

se détachent comme étant celles qui contribuent le plus fortement au calcul du score.

5 Limites et améliorations possibles

5.1 Performance intrinsèque

Le score AUC obtenu après la seconde itération pour la recherche des hyper paramètres optimaux de notre modèle est de 0,783.

Ce résultat est inférieur à celui de 0,792 obtenu par l'auteur du kernel 'LightGBM with Simple Features' proposé par 'A.Assklou _ Aguiar' et adapté ici pour la partie prétraitement. Cette différence s'explique par le fait que les auteurs de ce kernel ont optimisé l'ensemble des hyper paramètres en s'appuyant sur une optimisation Bayésienne qui pourrait également être mise en place ici.

D'autres méthodes d'optimisation peuvent être également testées, telles que l'utilisation de la librairie Hyper Opt.

De même, il est à noter que le score obtenu par 'A.Assklou _ Aguiar' reste, inférieur au meilleur résultat obtenu dans la compétition. Au moment de la rédaction du présent document, l'équipe 'Kraków, Lublin i Zhabinka' a obtenu un score de 0,817 avec un notebook public. Il serait donc possible de répliquer leurs méthodes de prétraitement, modélisation et d'optimisation pour dupliquer cette performance.

5.2 Performance métier

Les remarques précédentes étant posées, il convient de rappeler, comme abordé dans la section sur la mesure d'évaluation, que la sélection et l'optimisation du modèle s'appuient sur une mesure technique, le score AUC.

Afin de déterminer le seuil optimal d'un point de vue métier, la fonction 'gain' a été construite. Cette même fonction pourrait être utilisée pour la recherche d'optimisation des hyper paramètres avec la fonction 'make_scorer'.

5.3 Hypothèse métier

La fonction gain utilise une série d'hypothèses globales extérieures faisant appel :

- au chiffre d'affaires moyen par client,
- au gain moyen en l'absence de défaut de paiement et
- à la perte moyenne dans le cas d'un défaut de paiement.

Ces hypothèses peuvent être affinées dans un premier temps en utilisant les valeurs moyennes constatées dans le jeu de données d'entraînement. Dans un second temps, il est possible de d'utiliser des valeurs moyennes qui caractériseraient par exemple des classes de variables pour les 4 variables les plus importantes.

De façon plus absolue, les profits et pertes pourraient être calculé de façon individuelle.