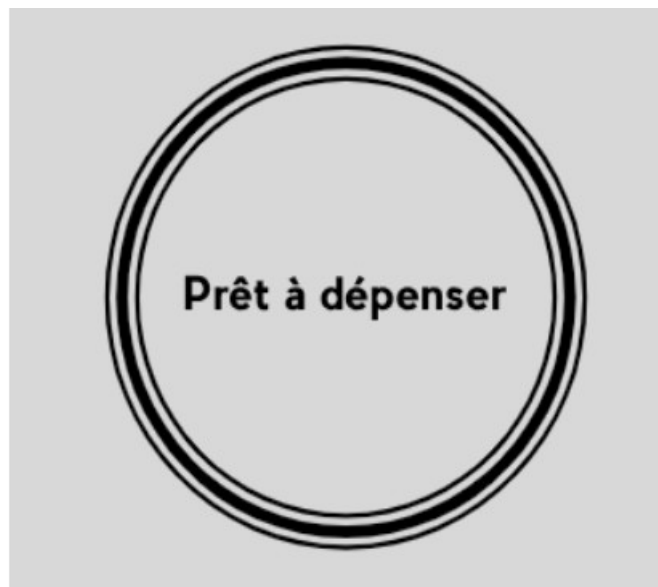


# **NOTE MÉTHODOLOGIQUE**

## **DÉMARCHE DE MODÉLISATION**



## TABLE DES MATIÈRES

Introduction.....	3
Le Projet.....	3
Démarche de modélisation.....	4
Entraînement du modèle.....	4
Fonction coût métier, l'algorithme d'optimisation et métrique d'évaluation.....	5
Fonction Coût.....	5
Algorithme d'optimisation.....	6
Métrique d'évaluation.....	7
L'interprétabilité du modèle.....	8
Feature Importance.....	8
SHAP.....	9
Les limites et les améliorations envisageables, performance et interprétabilité.....	10
Annexe.....	11

## INTRODUCTION

Cette note méthodologique est issue du projet 7 d'openclassrooms de data science intitulé "Implémentez un modèle de scoring"

## LE PROJET

La société "Prêt à dépenser", propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt. L'entreprise souhaite mettre en œuvre un outil de "scoring crédit".

La modèle à créer consiste en un modèle de scoring, fournissant une probabilité. Cette probabilité représente le risque de défaut du client lors de son crédit.

Des modélisations déjà existantes sont disponibles depuis la plateforme Kaggle. Les données collectées issues de cette même plateforme représentent 300 000 clients. Une centaine de variables décrit chacun des clients (score issu d'outil extérieur, revenus, durée du crédit, etc.)

### ENTRAÎNEMENT DU MODÈLE

Le jeu de données a été séparé en un jeu d'entraînement et de test (80%/20%) afin de réduire le risque sur-apprentissage.

La cible de ce modèle est distribuée de façon assez déséquilibrée (92%/8%), un équilibrage des données a été réalisé grâce à la librairie SMOTE.

Les hyperparamètres ont été optimisés via la méthode RandomizedSearchCV, les hyperparamamètres sont ceux de la librairie SMOTE ainsi que ceux de l'algorithme en tant que tel.

Les algorithmes étudiés sont ceux de la famille des modèles ensemblistes (random forest, light gbm). Une référence a été prise avec un modèle naïf.

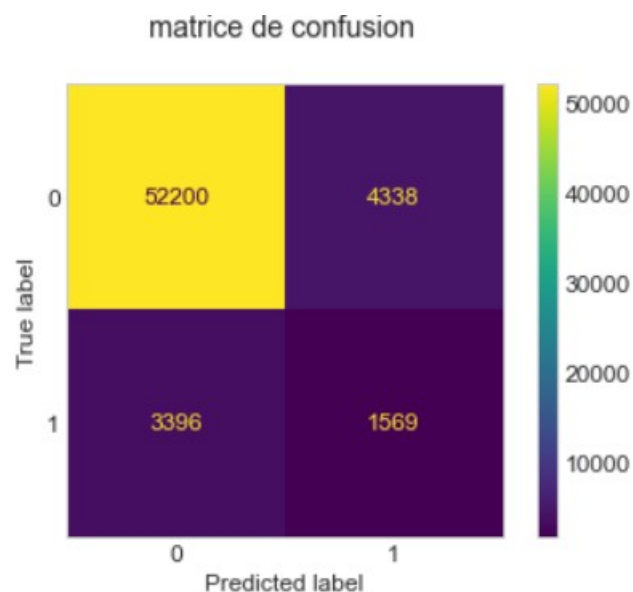
## FONCTION COÛT

Une fonction coût a été mise en place à partir des valeurs “vrai positif”, “vrai négatif”, “faux positif” et “faux négatif”.

Ces valeurs peuvent être représentées par une matrice de confusion.

Plusieurs combinaisons – en pondérant ces valeurs – ont été testées. Ici, on a cherché à minimiser les erreurs produites sur les prédictions des clients pouvant vraiment faire défaut. Au final, on a minimisé la valeur faux négatif + vrai négatif.

Ce coût est programmé dans une fonction et peut être amené à évoluer si le client souhaite réaliser un arbitrage différent pour sa prise de décision.



L'algorithme utilisé est l'algorithme de boosting lighGBM.

Il s'agit d'un modèle ensembliste permettant d'intégrer des variables catégorielles sans encodage "one hot encoder". Le temps de calcul est optimisé par rapport aux algorithmes le précédant. Des options existent aussi pour réaliser du calcul distribué.

1. **learning\_rate**
2. **max\_depth**: default is 20. Important to note that tree still grows leaf-wise. Hence it is important to tune **num\_leaves** (number of leaves in a tree) which should be smaller than  $2^{(\text{max\_depth})}$ . It is a very important parameter for LGBM
3. **min\_data\_in\_leaf**: default=20, alias= min\_data, min\_child\_samples

**Paramètres contrôlant le surapprentissage**

On étudie un problème de classification binaire, on peut donc utiliser comme métrique les valeurs de roc-auc (aire sous la courbe), de précision, de sensibilité ainsi qu'une matrice de confusion. La métrique d'optimisation utilisée est celle décrite précédemment pour la fonction coût.

Une métrique observée est la matrice de confusion :

		Classe réelle	
		-	+
Classe prédite	-	<b>True Negatives</b> <i>(vrais négatifs)</i>	<b>False Negatives</b> <i>(faux négatifs)</i>
	+	<b>False Positives</b> <i>(faux positifs)</i>	<b>True Positives</b> <i>(vrais positifs)</i>

Matrice de confusion

Les autres métriques :

Avec TN: True Negatives, TP: True Positives, False Negatives: FN, False Positives: FP.

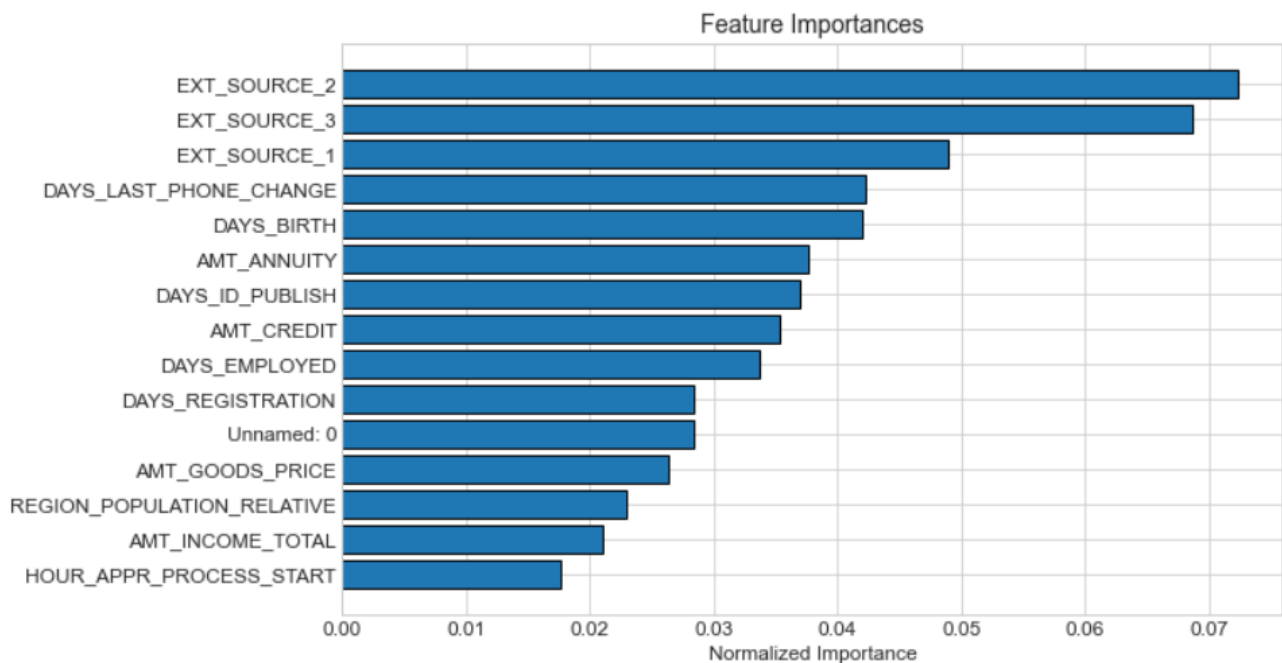
- $\text{accuracy} = (TN+TP) / (TN+TP+FN+FP)$ , proportion d'erreur globale
- $\text{précision} = TP/(TP+FP)$ , proportion de prédictions correctes parmi les points prédits positifs
- $\text{sensibilité} = TP/(TP+FN)$ , proportion de positifs correctement identifiés

L'évaluation actuelle du modèle donne les valeurs suivantes:

- $\text{accuracy} \sim 0,75$
- $\text{précision} \sim 30 \%$
- $\text{sensibilité} \sim 30 \%$

## FEATURE IMPORTANCE

Les variables influentes principales du modèle sont représentées dans le graphique ci-dessous:



A partir de ces variables, on peut déduire une première liste de variables intéressantes pour la création du tableau de bord, correspondant à des notions métiers :

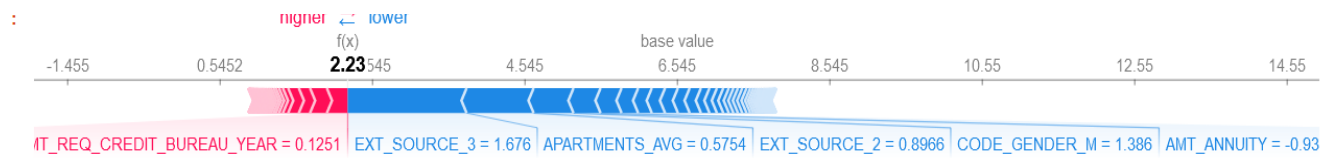
- EXT\_SOURCE\_ : score provenant d'un calcul externe
- AMT\_ANNUITY : annuité
- AMT\_CREDIT : montant du crédit
- DAYS\_EMPLOYED : nombre de jours en emploi
- AMT\_GOODS\_PRICE : montant du bien
- AMT\_INCOME\_TOTAL : ressources du client



## SHAP

La librairie shap permet d'étudier l'interprétation locale (et globale) du modèle.

Par exemple localement, pour un client, on peut visualiser dans quelle mesure les variables contribuent au calcul de son score particulier (ici les annuités, le score externe 3, etc.):



Ce modèle présente des performances en termes de sensibilité de 30 % et de 75 % d'accuracy. Cela permet d'offrir une première orientation dans l'évaluation du risque d'octroi de crédit.

Une voie possible d'amélioration est le feature engineering qui est assez complexe à réaliser de part le manque de transparence des variables extérieures et du nombre de variables en jeu.

Une alternative est de se documenter sur kaggle pour trouver un feature engineering plus abouti.

Une autre voie d'amélioration est la sélection de modèle compatible avec ce type de donnée comme des réseaux de neurones, qui pourrait permettre de simplifier l'étape de création de variables.

Le point essentiel, ici, est de pouvoir bénéficier du retour d'expérience de personnes du métier afin de définir correctement les bonnes variables et algorithmes. Il peut s'agir d'une première étape afin de converger avec le client vers des modèles plus performants en capacité de prédiction.

**Librairie SMOTE:** (<https://imbalanced-learn.org/>)

Lemaître, Guillaume, Fernando Nogueira, et Christos K. Aridas. « Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning ». *Journal of Machine Learning Research* 18, n° 17 (2017): 1-5.

<http://jmlr.org/papers/v18/16-365>

**Librairie SHAP:** (<https://shap.readthedocs.io/>)

Lundberg, Scott M, et Su-In Lee. « A Unified Approach to Interpreting Model Predictions ». In *Advances in Neural Information Processing Systems 30*, édité par I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, et R. Garnett, 4765-74. Curran Associates, Inc., 2017.

<http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf>.

**LightGBM:** (<https://lightgbm.readthedocs.io/>)

Ke, Guolin, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, et Tie-Yan Liu. « LightGBM: A Highly Efficient Gradient Boosting Decision Tree ». In *Advances in Neural Information Processing Systems*, édité par I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, et R. Garnett, Vol. 30. Curran Associates, Inc., 2017.

<https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf>.