Note méthodologique

Projet 7 - Parcours Data Scientist

“ Implémentez un modèle de scoring”

A.HAMEG , juin 2022

1. **Résumé :**

Cette note présente le processus de modélisation et d'interpérabilité du modèle mis en place.

L'entreprise "Prêt à dépenser" (*société de crédit à la consommation*) souhaite construire un modèle de scoring qui donne une prédiction sur la probabilité de faillite d'un client de façon automatique.

Cela se présente sous forme d'un dashboard interactif à destination des gestionnaires de la relation client permettant d'interpréter les prédictions faites par le modèle, et d'améliorer la connaissance client des chargés de relation client.

la base de donnée contient 307.000 clients avec 121 features (revenus, âge, sexe, emploi, logement, notations externes…)

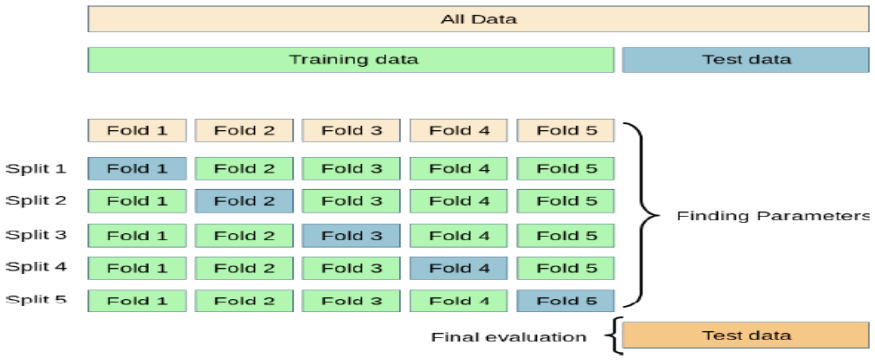
2. **Méthodologie d'entraînement du modèle**

Un kernel Kaggle a été utilisé (téléchargeable sur le site Kaggle) Après analyse et adaptation de ce dernier pour qu'il réponde aux besoins de la mission, le modèle utilisé a été entraîné dessus.

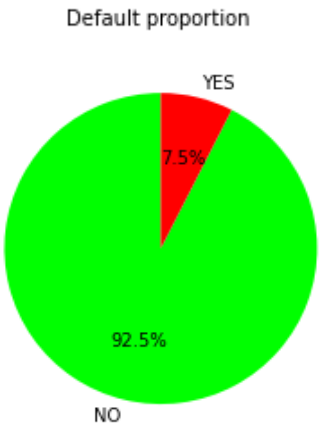
Le jeu de donnée a été splitté en deux parties :

1- jeu d'entraînement (70% des clients) : Ce dernier a été splitté en plusieurs folds pour entraîner différents modèles (DummyClassifier, LightGBM, RandomForest Classifier, Logistic Regression) et optimiser leurs paramètres (cross-validation) sans overfitting.

2- jeu de test (30% des clients) : Ce dernier a été utilisé pour évaluer les modèles testés.



Nous sommes dans ce cas confrontés à un problème de classification binaire d'une base de données avec des classes déséquilibrée

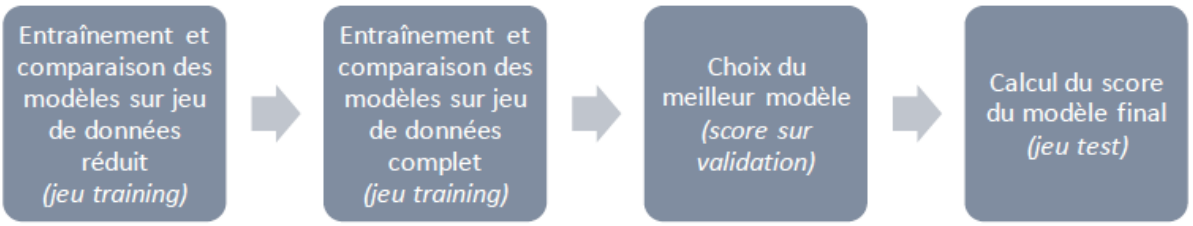


À cause de ce déséquilibre, un modèle naïf qui prédit que les clients sont sans défaut de paiement aurait une précision de 92,5%. Il pourrait donc être considéré à tort comme un modèle très performant alors que ce dernier ne détecte pas les clients à risques . Il faut donc rééquilibrer les classes.

Pour ce faire, la méthode “Class Weights” a été utilisée (la méthode SMOTE a également été testée sans être retenue (*temps de calculs élevé, overfitting..*)

- ☐ Class Weights permet de modifier les poids associés aux observations afin qu’une observation mal classée dans la classe minoritaire pénalise davantage la fonction coût qu’une observation mal classée dans la classe majoritaire.

10% du jeu de données initial a été utilisé pour comparer les deux approches.
Le processus d’optimisation a ensuite été appliqué à l’ensemble du jeu de données.
Le meilleur modèle est celui qui a obtenu le meilleur score sur jeu de validation.



3. La fonction coût, l'algorithme d'optimisation et la métrique d'évaluation

3.1 Fonction coût : La fonction coût prend en compte la distribution des observations dans le jeu de données d'entraînement, L’apprentissage adopté est un apprentissage « sensible au coût », il se traduit par une optimisation des

hyperparamètres **scale_pos_weight** et **class_weight**. Suite à différents essais et à des temps de calculs assez importants, le ratio a été figé à 1:20 (légèrement supérieure au ratio réel 1:8).

3.2 Métrique d'évaluation : La matrice de confusion est la suivante :

		Prédit	
		Client en défaut	Client sans défaut
Réel	Client en défaut	TP	FN
	Client sans défaut	FP	TN

Il est nécessaire de bien choisir la métrique d'évaluation. Il faut que cette métrique soit d'abord cohérente avec les objectifs du métier mais aussi pertinente pour répondre à sa problématique :

$$Précision = \frac{VP}{VP + FP} \quad Rappel = \frac{VP}{VP + FN} \quad F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \quad Beta = \frac{coefficient\ Recall}{coefficient\ precision}$$

Contenu du contexte métier, 2 types de coûts financiers (pertes) sont possibles pour l'entreprise :

1/ mal catégoriser un client avec un risque élevé de défaut (Faux négatif = erreur de type II)

2/ mal catégorisé un client avec un risque faible de défaut (Faux positif = erreur de type I)

Nous souhaitons donc minimiser les faux négatifs et les faux positifs tout en maximisant les vrais positifs. Autrement dit, on cherche à maximiser le recall et la précision.

Dans notre cas, le FN a un coût plus élevé que le FP. Perdre un client potentiel coûte moins cher que d'attribuer un crédit à un client qui fera défaut de remboursement.

Nous cherchons donc une fonction qui optimise les deux critères tout en donnant plus d'importance au Rappel. Par conséquent, le choix de la fonction F-Beta score a été fait. Le coefficient Bêta représente

l'importance d'une métrique par rapport à l'autre (Recall/précision). La valeur de Bêta a été calculée en considérant les hypothèses métiers suivantes :

1. Le coût d'un FN représente 20% du montant moyen de crédit attribué aux personnes en défaut.
2. Le coût d'un FP représente 5% du montant moyen de crédit refusé aux personnes sans défaut.

Cela donne un Recall = 15% et une précision = 5% pour un ratio de 7.5%. => 2.775

3.3 Algorithme d'optimisation : Une RandomizedSearchCV a été utilisée pour l'optimisation des hyperparamètres. Le modèle ayant le meilleur score (F–Beta score) a été retenu/ Il s'agit du LightGBM Classifier.

4. interprétabilité globale et locale du modèle

Le dashboard est destiné à des gestionnaires de la relation client. Ces derniers doivent être capables d'expliquer au client les décisions prises par le modèle. Pour cela, la méthode SHAP a été utilisée afin de comprendre comment chacune des variables du client à influencer sur la décision finale du modèle.

5. Axes d'amélioration :

- Définir avec plus de précision en collaboration avec le métier, la métrique d'évaluation
- Travailler plus l'analyse Exploratoire des données (ajout de nouvelles features, nettoyage plus précis des données en s'appuyant sur la connaissance métier.