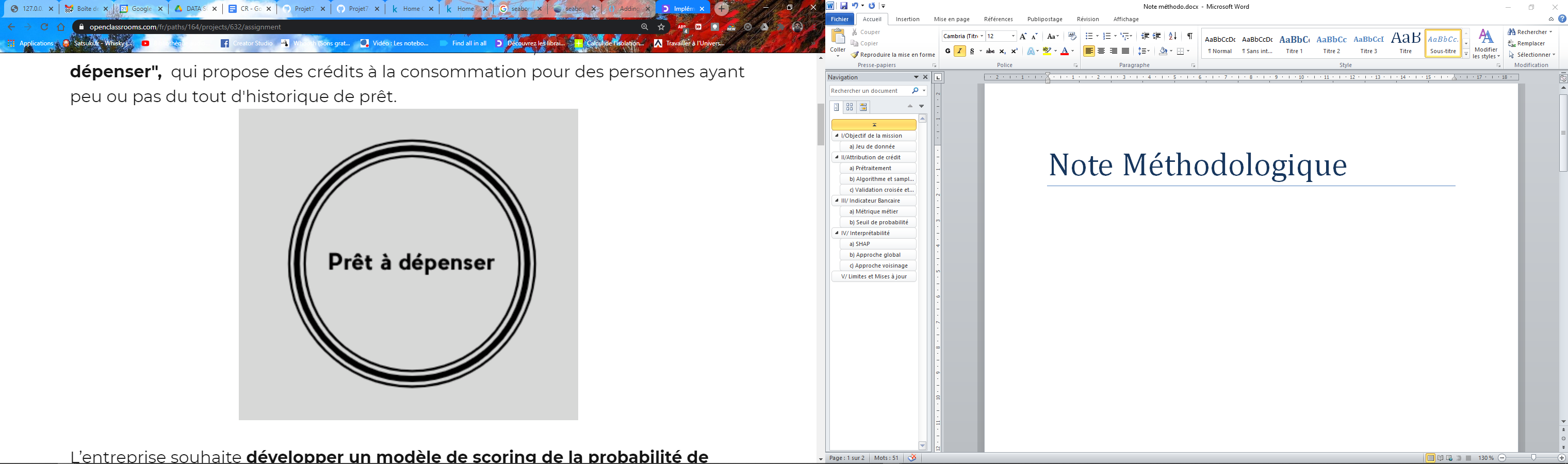
Note Méthodologique

**Implémentez un modèle de scoring**



Mai 2022

**Projet 7 - Data Scientist - OpenClassroom**

**Nhat Tam VO**

Table des matières

[I. Contexte 3](#_Toc104238806)

[II. Jeu de donnée 3](#_Toc104238807)

[III. Démarche de modélisation 3](#_Toc104238808)

[IV. Fonction coût, algorithme d’optimisation et métrique d’évaluation 3](#_Toc104238809)

[V. Interprétabilité du modèle 3](#_Toc104238810)

[VI. Limites et améliorations 3](#_Toc104238811)

1. **Introduction**

Ce projet consiste à développer pour la société financière « Prêt à dépenser », un outil pour aider la décision d'accorder ou non un prêt à un client potentiel en s’appuyant sur des sources de données variées (données comportementales, données provenant d'autres institutions financières, etc.).

Deux objectives de ce projet est de développer un modèle de scoring de la probabilité de défaut de paiement d’un client et de développer un Dashboard interactif pour assurer une transparence sur les décisions d’octroi de crédit

1. **Jeu de donnée**

Un kernel Kaggle existant a été utilisé pour faciliter la préparation des données nécessaires.[[1]](#footnote-1) Les données traitées utilisée pour ce projet sont une base de données de 307 000 clients comportants 363 features (âge, sexe, emploi, logement, revenus, informations relatives au crédit, notation externe, etc.).

Dans ce projet, Nous sommes dans problème de classification binaire à deux classes (0 ou 1). Cependant, il apparaît un fort déséquilibre entre non défaut de paiement (classe 0) et défaut de paiement (classe 1). Ce constat est représenté sur la figure 1.

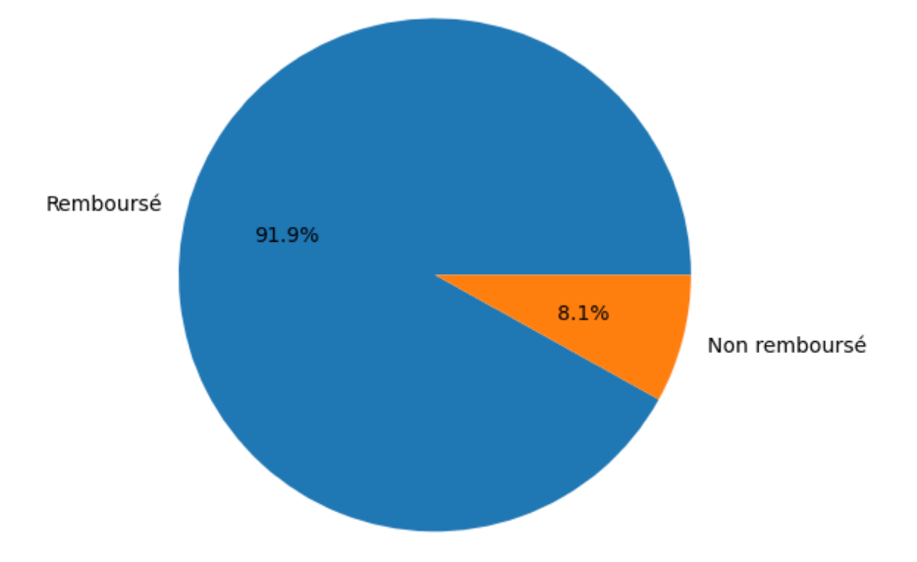


Figure 1 : Jeu de donnée déséquilibre

Ce déséquilibre des classes doit être pris en compte dans l’entraînement des modèles puisqu’un modèle naïf prédisant systématiquement classe 0 pour tous les cas aurait score de l’accuracy de 0.92. Avec ce accuracy, ce modèle naïf pourrait être considéré comme un modèle performant et fiable mais sa prédiction n’est pas vraiment efficace et fiable. Pour réduire la conséquence de ce déséquilibrage, deux approches SMOTE et Class Weights sont été utilisées dans ce projet. Class Weights est une méthode permettant de pénaliser les poids associés aux observations sur les représentées. SMOTE ou over-sampling permet de créer une nouvelle donnée synthétique à partir de donnée existante.

1. **Démarche de modélisation**

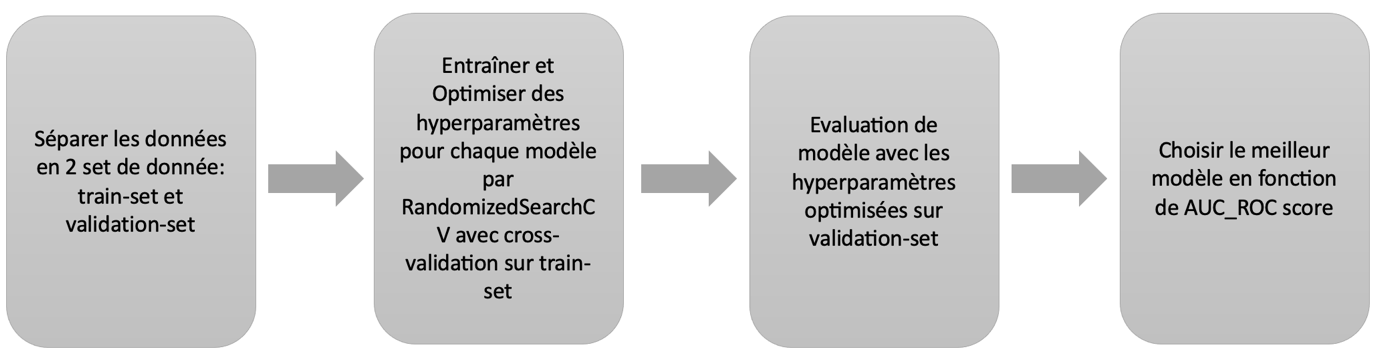


Figure 2 : Méthodologie d’entraînement du modèle

La méthodologie d’entraînement du modèle est montré sur la figure 2. Tout d’abord, le jeu de données initial a été séparé́ en deux parties : un jeu de training contenant 70% des individus et un jeu de validation contenant 30 % des individus. Le jeu de training a été utilisé pour entraîner les différents modèles et optimiser les hyperparamètres des modèles en utilisant RandomizedsearchCV with Cross-validation. Le jeu de validation a été utilisé pour évaluer du modèle entrainé avec les hyperparamètres optimisées trouvées.

Le meilleur modèle a été choisi en fonction de score AUC\_ROC (aire sous la courbe de la caractéristique d'exploitation du récepteur). Cette courbe trace le taux de vrais positifs en fonction du taux de faux positifs. Cette métrique est comprise entre 0 et 1, un meilleur modèle obtenant un score plus élevé. Un modèle qui se contente de deviner au hasard aura une AUC ROC de 0,5 (Figure 3). Cette métrique a été utilisé dans ce projet car elle est adaptée dans les cas où les classes sont déséquilibré comme ici. En plus, elle mesure la performance globale d’un modèle de classification pour tous les seuils de classification. Donc que, le meilleur modèle choisi avec AUC\_ROC n’est pas impacté par un choix de seuil.

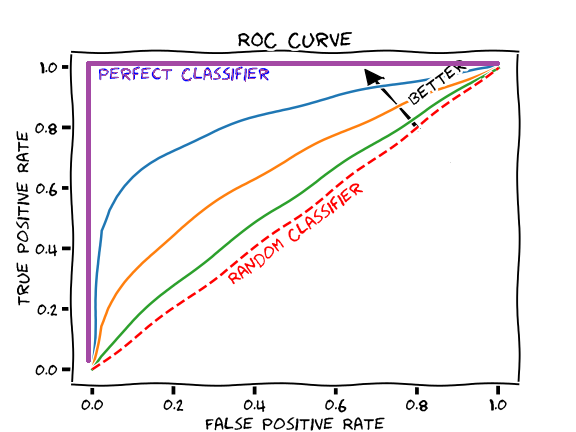


Figure 3 : Courbe ROC des modèles avec différents performances

J’ai également évalué les modèles par le temps d’exécution et d’entraînement. 7 modèles ont été entraînés et optimisés pour ce problème de classification. DummyClassifier a été utlisé comme baseline. Les scores AUC\_ROC de ces modèles optimisés et leurs temps d’exécution sont présentés dans la figure 4 suivant :

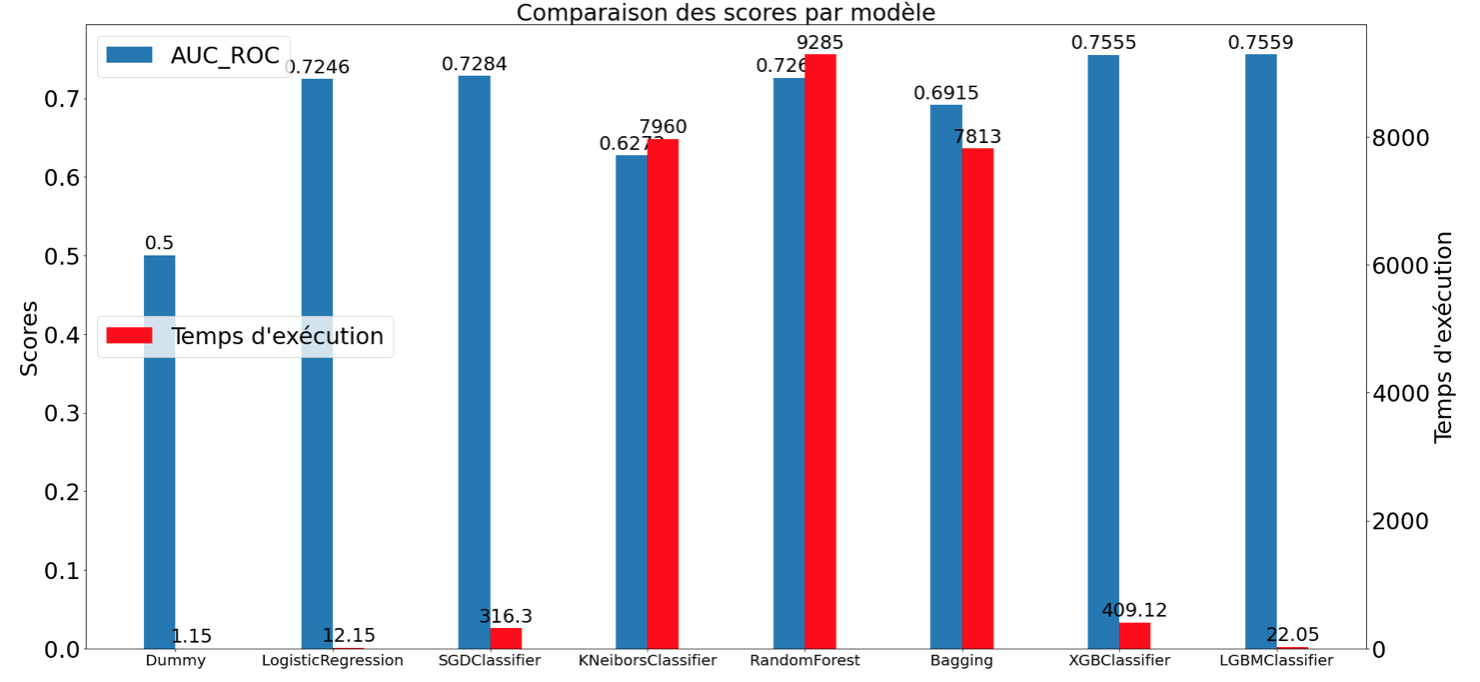


Figure 4 : Comparaison des scores par modèle

En définitive, le modèle le plus performant retenu est LGBMClassifier en regard de son élevé score AUC\_ROC et son rapide temps d’exécution. Learning curve et validation cuvre qui ont été ensuite réalisés, montrent que ce modèle optimisé (n\_estimators = 150, learning\_rate = 0.06) entraîné par cette quantité de donné n’est pas dans le cas de l’over-fitting (Figure 5).

|  |  |  |
| --- | --- | --- |
|  |  |  |

Figure 5 : (Gauche) Learning Curve

1. **Fonction coût, algorithme d’optimisation et métrique d’évaluation**

Afin de rendre le modèle le plus cohérent possible une nouvelle **métrique spécifique** à la mission est implémentée. La fonction métrique se construit alors en pénalisant des prédictions erronées et en gratifiant des prédictions correctes.

La matrice de confusion en sortie de prédiction est représentée dans le tableau suivant :

|  |  |  |
| --- | --- | --- |
|  | Clients prédits en défaut | Clients prédits sans défaut |
| Clients réels en défaut | Vrais positifs | Faux négatifs |
| Clients réels sans défaut | Faux positifs | Vrais négatifs |

Du point de vue d’une banque, on cherchera à éviter de considérer un mauvais client avec un fort risque de défaut de paiement comme un bon client. Un accord du prêt à un mauvais client conduit fortement une perte financière importante. On cherchera aussi à éviter de mal catégoriser un bon client car la banque aura perdu un gain d’argent potentiel à cause d’un crédit non accordé. On cherche donc à minimiser les pourcentages de faux négatifs et de faux positifs. Cependant, un faux positif n’a pas même coût qu’un faux négatif. Un faux négatif est environ 10 fois plus coûteux qu’un faux positif. Alors, la fonction de coût métier dans ce problème est de type 10\*FN + FP (FN : faux négatifs, FP : faux positifs). On cherche donc à minimiser cette fonction.

Avec le modèle LGBMClassifier choisi et optimisé, un seuil de classification de 0.55 a été trouvé en minimisant la fonction 10\*FN +FP (Figure 6)

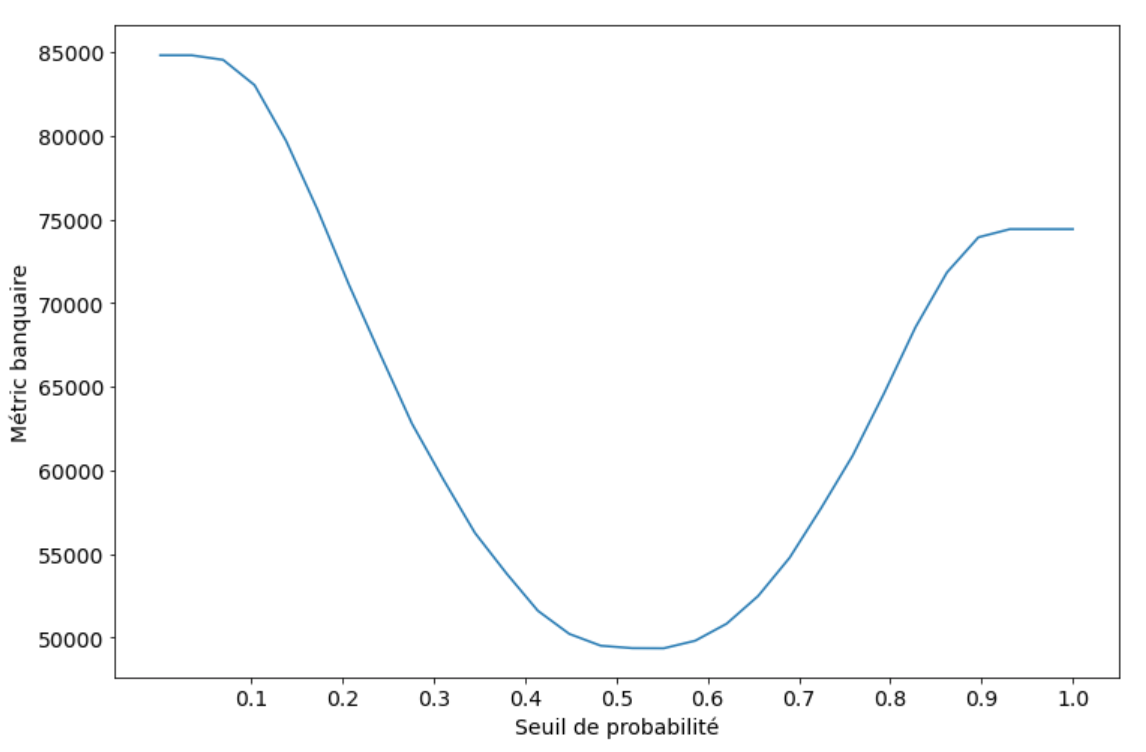


Figure 6 :Variation de fonction métier 10\*FN + FP en fonction de seuil de classification

1. **Interprétabilité du modèle**

Pour garantir la transparence auprès de ses clients, un dashboard interactif est réalisé afin de comprendre la décision d’attribution ou de refus de prêt.

La librairie SHAP utilisée est une librairie s’appliquant à n’importe quel modèle de machine learning et permettant de comprendre l’évolution des prédictions d’un modèle. Principalement, la prédiction du modèle est interprétée en deux niveau, global et local:

* L’interprétation globale permet de savoir quelles sont l’ensemble des caractéristiques importantes qui ont le plus influencés le model dans sa prédiction (Figure 6 – en haut).
* L’interprétation local montre précisément les caractéristiques importantes retenues par le modèle pour réaliser une prédiction de classe 0 ou classe 1. Par example, une prédiction de client non solvable (défaut de paiement) a été localement interprétée par la figure 6 – en bas. Pour ce client, les caractéristiques avec les valeurs positives de SHAP comme « EXT\_SOURCE\_3 » et « PAYMENT\_RATE » font augmenter le risque de défaut de paiement, tandis que les caractéristiques avec les valeurs négatives de SHAP comme « EXT\_SOURCE\_2 » font diminuer le risque de prêt de crédit.

|  |
| --- |
|  |
|  |

Figure 7 : (Haut) Interprétation global et (Bas) Interprétation local

1. **Limites et améliorations**

La partie de traitement préalable du jeu de données a été réalisée en réutilisant un notebook issu de Kaggle qui crée des nouvelle features par groupement de donnés en appliquant diverses fonctions comme Min, Max, Mean, etc. Il y a très probablement l’opportunité́ d’améliorer la modélisation qui permettra une meilleure performance prédictive au modèle, en utilisant d’autres features des données fournies, notamment les nouvelles features créées par la collaboration avec les équipes métier banquier.

Le score AUC\_ROC mesure de façon globale la performance d’un modèle de classification. Donc que la performance de modèle pourrait être améliorée en utilisant une autre métrique d’évaluation basé sur des hypothèses métier confirmées.

La recherche des meilleurs hyperparamètres ont été réalisés sur un nombre restreint d’hyperparamètre et sur un nombre restreint de valeur pour chaque hyperparamètre testé L’optimisation des modèles peut être aussi améliorée par tester avec plus nombre d’hyperparamètre et de valeurs des hyperparamètres.

Concernant le Dashboard interactif, les améliorations à réaliser pour la deuxième version doivent permettre de modifier les données de client et d’effectuer une nouvelle prédiction à base de nouvelle donnée modifiée. Il est aussi intéressant si la nouvelle version de Dashboard permet insérer les données d’un nouveau client et donner la prédiction pour ce nouveau client.

Enfin, le Dashboard pourra être amélioré sur les graphes interactifs pour répondre au mieux aux attentes et besoins des conseillers clients.

1. <https://www.kaggle.com/code/jsaguiar/lightgbm-with-simple-features/script> [↑](#footnote-ref-1)