Projet 7 Parcours Data Scientist

ROUIBAH KHAYREDDINE Novembre 2022 OpenClassrooms

Note Méthodologique

1. La méthodologie d'entraînement du modèle

Qu’est-ce que le machine learning ?

Le machine learning (ML) est une forme d’intelligence artificielle (IA) qui est axée sur la création de systèmes qui apprennent, ou améliorent leurs performances, en fonction des données qu’ils traitent. L’intelligence artificielle est un terme large qui désigne des systèmes ou des machines simulant une forme d'intelligence humaine. Le machine learning et l'IA sont souvent abordés ensemble et ces termes sont parfois utilisés de manière interchangeable bien qu'ils ne renvoient pas exactement au même concept. Une distinction importante est que, même si l’intégralité du machine learning repose sur l’intelligence artificielle, cette dernière ne se limite pas au machine learning.

Aujourd’hui, nous utilisons le machine learning dans tous les domaines. Lorsque nous interagissons avec les banques, achetons en ligne ou utilisons les médias sociaux, des algorithmes de machine learning entrent en jeu pour optimiser, fluidifier et sécuriser notre expérience. Le machine learning et la technologie qui l’entoure se développent rapidement, et nous commençons seulement à entrevoir ses capacités.

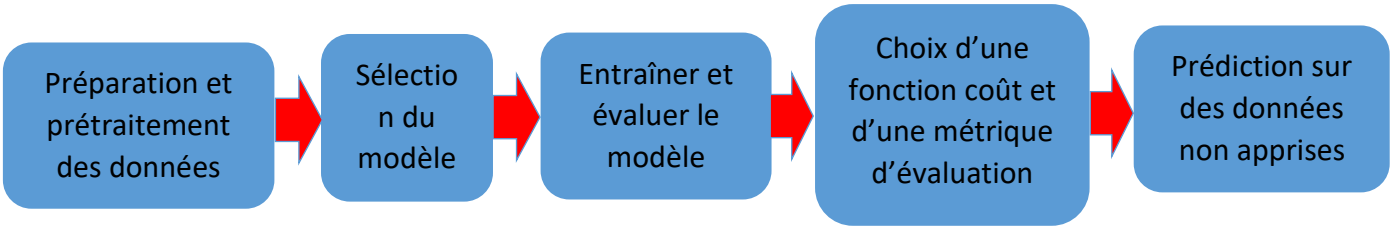
Approches de l'apprentissage ?

Les algorithmes sont les moteurs du machine learning. En général, deux principaux types d'algorithmes de machine learning sont utilisés aujourd'hui : **l'apprentissage supervisé** et **l'apprentissage non supervisé**. La différence entre les deux se définit par la méthode employée pour traiter les données afin de faire des prédictions.

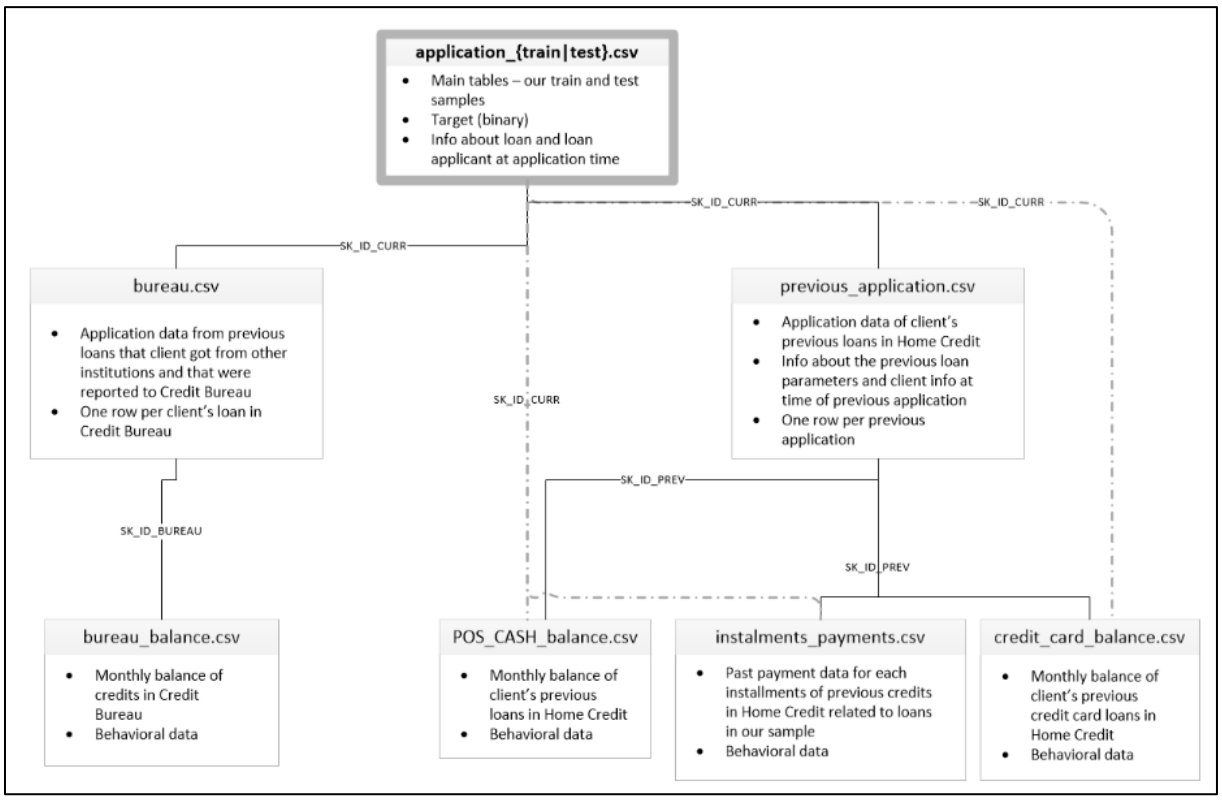
Dans notre cas-ci, les entrées de la fonction sont les informations des clients, tandis que leur solvabilité ou non représente la sortie du modèle, ainsi qu’on parle de l’apprentissage supervisé.

Un projet de Machine Learning comprend les étapes suivantes :

- Préparation et prétraitement des données
- Sélection du modèle
- Entraîner et évaluer le modèle
- Choix d’une fonction coût et d’une métrique d’évaluation
- Prédiction sur des données non apprises



Préparation et prétraitement des données



Le prétraitement consiste à nettoyer le jeu de données, en traitant les valeurs aberrantes et les variables dont les valeurs manquantes NaN ou bien le taux de remplissage est trop faible.

Puis encoder les variables catégorielles afin de rendre ces variables interprétables. Cette étape se suit par le « Features Engineering » une étape qui consiste à trouver une bonne sélection de Features pertinentes parmi celles qui existent dans le but d'améliorer la performance du modèle.

Sur l'image au-dessus représente l'ensemble des données d'entrée 7 fichiers CSV, et pour l'assemblage des Data-Set on a utilisé le Karnel Kaggle, ce Karnel a été modifié et commenté pour qu'il soit adapté à notre besoin.

Karnel Kaggle :

- Il faut savoir que ce Karnel a utilisé les mêmes Data-Sets sauf le fichier « **test.csv** » ce dernier ne contient pas de cible « Target » qui sera utilisé par la suite comme nouveaux clients sachant qu'il contient les mêmes autres Features que « **train.csv** »

- Concernant l'encodage des variables catégorielles on constate deux types d'encodage, le premier il s'agit de **Label Encoding** dédié pour les variables à deux catégories, et le deuxième c'est pour les variables de plus d'une catégorie le **One Hot Encoding,** il y a également une étape de correction de modalités.

- Les valeurs aberrantes ont été traitées également en prenant l'exemple de « DAY_EMPLOYED », ces valeurs ont été remplacées par des valeurs NaN qui seront remplacé par la suite par la médian ou bien imputée, il y a aussi des valeurs « inf » remplacé également par des valeurs NaN

```
# NaN values for DAYS_EMPLOYED: 365.243 -> nan
df['DAYS_EMPLOYED'].replace(365243, np.nan, inplace= True)
```

- Le traitement des valeurs manquantes est passé par deux étapes, la première consiste à éliminer les variables de plus de 50% de valeurs manquantes, puis l'imputation par la médiane

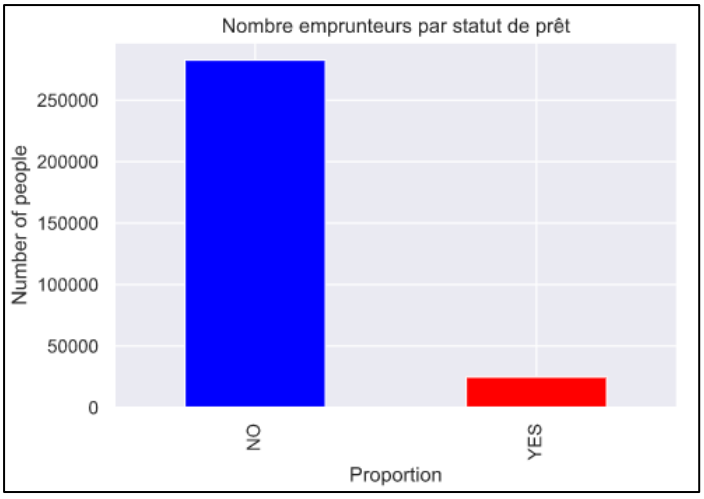
- Il y a également une étape de Features Engineering, une série de variables a été supprimée car ces variables ont une forte corrélation avec d'autres variable ou bien une faible corrélation avec la cible, ou l'ajout de nouvelles variable calculées et ajoutées manuellement comme par exemple la variable PAYMENT_RATE :

```
df['PAYMENT_RATE'] = df['AMT_ANNUITY'] / df['AMT_CREDIT']
```

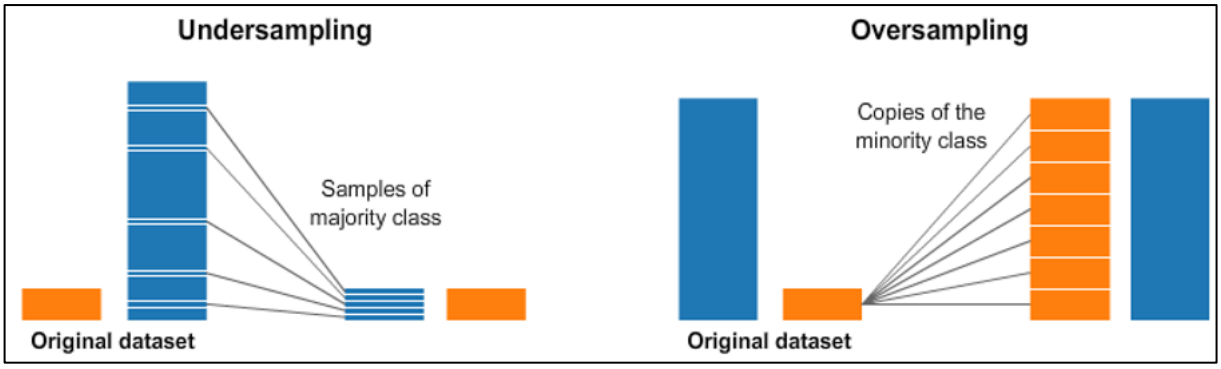
RESAMPLING

Le grand souci rencontré dans la classification supervisée c'est le déséquilibre des cibles « Target », dans notre cas on parle bien d'une classification binaire dans laquelle la classe 0 représente les personnes qui ont remboursé leur prêt et la classe 1 sont les personnes qui ont eu des difficultés à rembourser leur prêt.

C'est le cas pour cette modélisation, on a de la classe 0 qui domine si on entraine le modèle sur ces données brutes on aurait forcément une prédiction très favorite pour la classe 0.



Donc, pour éviter ce problème il existe plusieurs techniques présentent dans la librairie **Imblearn**. La technique utilisée est le Le sur-échantillonnage synthétique SMOTE



- **Le sous-échantillonnage** aléatoire (random undersampling) des observatio ns majoritaires : on retire aléatoirement des observations majoritaires
- **Le sur-échantillonnage** aléatoire (random oversampling) des observations minoritaires : on tire au hasard des individus minoritaires que l'on rajoute aux données.
- **SMOTE** : Le sur-échantillonnage synthétique (SMOTE pour Synthetic Minor ity Oversampling Technique) produit des observations minoritaires ressem blantes mais distinctes de celles déjà existantes.
- **SMOTETomek** : l'idée est de combiner SMOTE avec une technique de sous-échantillonnage (ENN, Tomek) pour augmenter l'efficacité de la gestion de la classe déséquilibrée.

```
1 X_train_sm, y_train_sm = resampling (X_train, y_train, sm)

Original dataset shape Counter({0: 212011, 1: 18619})
Resampled dataset shape Counter({0: 212011, 1: 212011})
```

Sélection du modèle entrainement

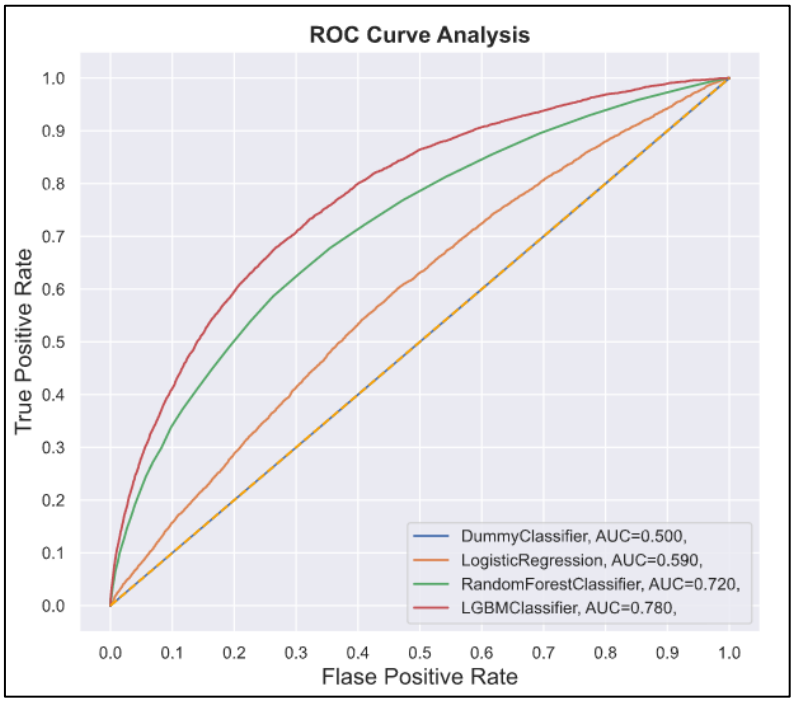
En machine learning il existe plusieurs modèles (Algo), chacun et ces caractéristiques sa performance, sa complexité, pour sélectionner le bon modèle on a sélectionné 4.

- DummyClassifier
- LogisticRegression
- RandomForestClassifier
- LGBMClassifier

L'entrainement s'applique sur le Data-Set « train.csv » généré après l'assemblage, et le prétraitement puis échantillonné avec la fonction « Train_Test_Split », 75% pour le training et 25% pour la validation.

On a testé ces quatre modèles avec leurs paramètres par défaut, pour avoir une première aperçue sur leurs performances sur les données d'entrées. Par la suite on les a classés par ordre comme illustré dans le tableau suivant et la courbe ROC :

	models	fpr	tpr	fprOpt	tprOpt	Accuracy	Precision	F_beta_Score	Recall	Time	AUC
3	LGBMClassifier	[0.0, 0.0, 0.0, 1.4150075702905011e-05, 1.4150...	[0.0, 0.00016113438607798906, 0.00048340315823...	0.2695	0.6819	0.92	0.60	0.04	0.03	20.24	0.78
2	RandomForestClassifier	[0.0, 0.0, 0.0, 2.8300151405810022e-05, 4.2450...	[0.0, 0.00016113438607798906, 0.00048340315823...	0.3533	0.6761	0.92	0.60	0.00	0.00	294.75	0.72
1	LogisticRegression	[0.0, 1.4150075702905011e-05, 0.00014150075702...	[0.0, 0.0, 0.0, 0.00016113438607798906, 0.0001...	0.47	0.5838	0.92	0.13	0.00	0.00	13.39	0.57
0	DummyClassifier	[0.0, 1.0]	[0.0, 1.0]	0.0	0.0	0.92	0.00	0.00	0.00	0.28	0.50



Le modèle LGBMClassifier était le meilleur au niveau de la performance soit au niveau du **Recall** ou bien de **la précision** ainsi que la courbe ROC l'air sous la courbe montre bien l'écart en l'ensemble des modèles**.**

Donc ce modèle a été sélectionné et qui est par la suite entrainé encore une fois avec **Grid Search CV** « la validation croisée » afin de trouver la bonne combinaison d'hyperparamètres.

Après l'entrainement ce modèle a été enregistré avec ces hyperparamètres qui sera après utilisé pour faire du déploiement de l'application

La fonction coût métier la métrique d'évaluation

Pour juger si un modèle est dit meilleur qu'un autre lorsqu'il donne des meilleurs résultats suivant une métrique de performance qui correspond à un cas spécifique. Pour ce projet, une fonction coût bien précise a été développée dans l'objectif de maximiser les gains obtenus par validation ou non des demandes de prêts bancaires. La fonction coût étant à maximiser pour ce problème.

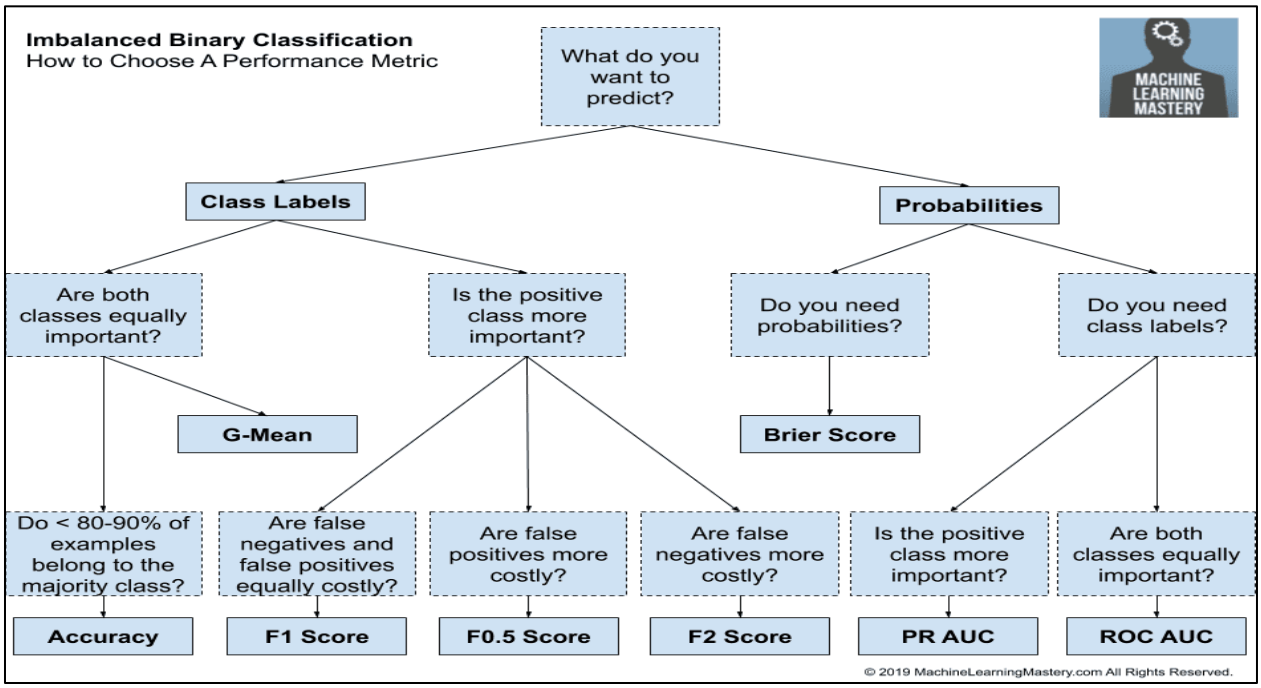
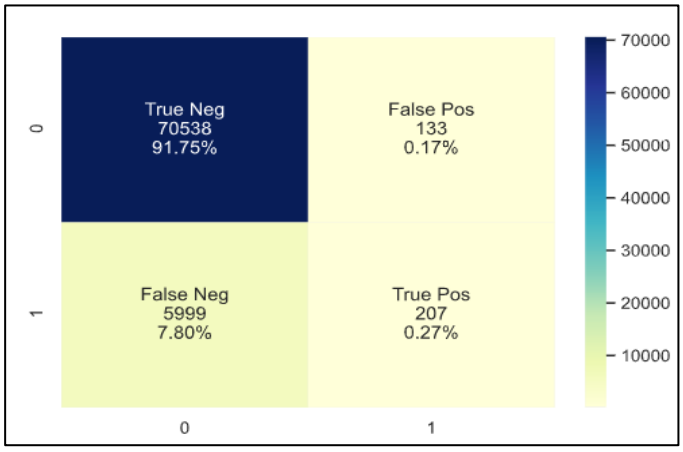


Figure : Table de Métriques

Il s'agit bien d'une classification binaire, 0 c'est négatif mais signifie que le prêt est solvable, 1 signifie que le prêt est non solvable donc voici les combinaisons possibles :

- TN : True Negatif, le modèle prédit 0 et la valeur réelle est 0
- TP : True Positif, le modèle prédit 1 et la valeur réelle est bien de 1
- FN : False Negatif, le modèle prédit 0 alors que la valeur réelle est bien de 1
- FP : False Positif, le modèle prédit 1 alors que la valeur réelle est de 0

Cette combinaison est présentée par la matrice de confusion : un tableau montrant les prédictions correctes et les types de prédictions incorrectes.



Parmi les métriques qui nous intéressent ici sont la Précision, et le Recall ce qui a été pris en compte lors de la sélection des modèles :

- **La Précision** : le nombre de vrais positifs divisé par toutes les prédictions positives. La précision est également appelée valeur prédictive positive. C'est une mesure de l'exactitude d'un classificateur. Une faible précision indique un nombre élevé de faux positifs.

```
============================================================
AUC : 0.7749
============
              precision    recall  f1-score   support

           0       0.97      0.65      0.78     70671
           1       0.16      0.76      0.26      6206

    accuracy                           0.66     76877
   macro avg       0.56      0.71      0.52     76877
weighted avg       0.90      0.66      0.74     76877
```

- **Le Recall** : le nombre de vrais positifs divisé par le nombre de valeurs positives dans les données de test. Le rappel est également appelé sensibilité ou taux de vrais positifs. C'est une mesure de l'exhaustivité d'un classificateur. Un faible rappel indique un nombre élevé de faux négatifs.

- **La fonction coût métier**

Le modèle peut se tromper de deux manières différentes, soit en prédisant positif alors que l'individu est négatif (False Positif) soit en prédisant négatif alors que l'individu est positif. En revanche, la perte d'argent est plus conséquente pour un FN (prêt accordé alors que le client n'est pas solvable) qu'un manque à gagner pour un FP (prêt non accordé alors que le client est solvable). Une fonction coût personnalisée a été créée pour tenir compte le risque et pénaliser surtout les FN (False Negative).

Sachant que la métrique Banking ici crée, consiste à calculer le gain obtenu pour l'ensemble des individus du jeu de données. Pour cela nous avons fixé des poids arbitraires pour chacune des prédictions relativement à leurs valeurs réelles.

Les valeurs des poids sont : **TP_value : 1, FN_value : -30, TN_value : 1, FP_value : -1**

seuil_optim_lgbm = 0.48

Des valeurs -30 et -1 ont été données arbitrairement pour les prêts prédit solvable FN, et non solvables FP dans le but de les pénaliser par contre les prêts solvables TN et non solvables TP dotés pour les valeurs de 1 et 1. On peut changer ces valeurs à la convenance du besoin métier.

Ces valeurs de coefficients signifient que les Faux Négatifs engendrent des pertes 10 fois plus importantes que les gains des Vrai Négatifs.

- **Seuil de solvabilité :**

Le modèle retourne un score entre 0 et 1 et par défaut il attribue la classe 1 lorsque le score est supérieur à 0.5 et 0 sinon seuil fixer par défaut. Il a été décidé d'optimiser ce seuil qui permet de définir si un client est solvable ou non.

Le seuil déterminé c'était calculé avec trois métriques différentes :

- **G-MEAN**

  Sensitivity = True Positive / (True Positive + False Negative)

  Specificity = True Negative / (False Positive + True Negative)

  Où : Sensitivity = True Positive Rate

  Specificity = 1 – False Positive Rate

  Geometric Mean ou G-Mean est une métrique pour les classes déséquilibrées qui permet d'optimiser en faisant un équilibre entre la sensibilité et la Spécificité.

  G-Mean = sqrt (Sensitivity * Specificity)

```
============================================================
thresholdOpt=0.53 gmean= 0.71 fprOpt = 0.31 tprOpt = 0.73
============================================================
```

- **Youden's J statistic :** youdenJ = tpr – fpr

```
============================================================
Best Threshold: 0.5343 with Youden J statistic: 0.7089
FPR: 0.3129, TPR: 0.7314
============================================================
```

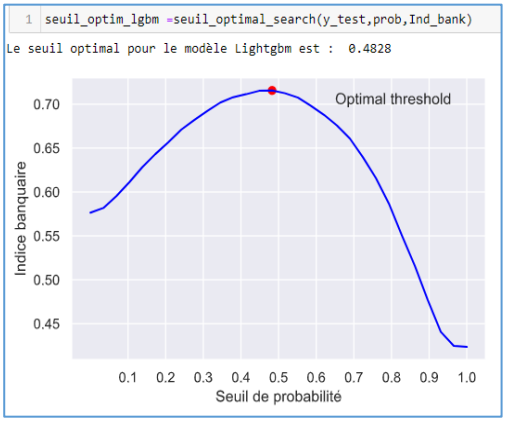
- **Fbeta-Measure**

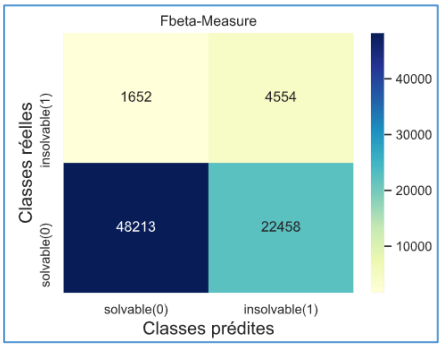
  Fbeta La mesure est F Résumé des mesures, Parmi eux Moyenne harmonique L'équilibre entre la précision et le taux de rappel dans le calcul est déterminé p ar beta Contrôle des coefficients.

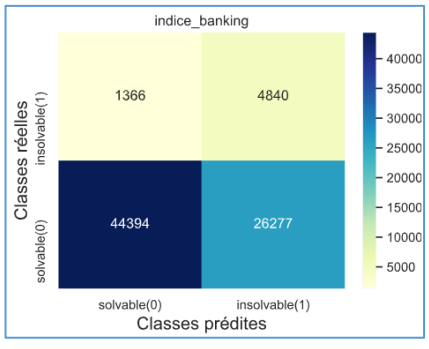
  Fscore = (5 * precision * recall) / (4 *precision + recall)

```
============================================
Best Threshold = 0.534333, F-Score = 0.441
============================================
```

On constate que les trois seuils sont les mêmes en arrondissant les chiffres, par contre, on a utilisé la **fonction de coût personnalisé** et le seuil calculé est de **seuil_optim_lgbm = 0.48**





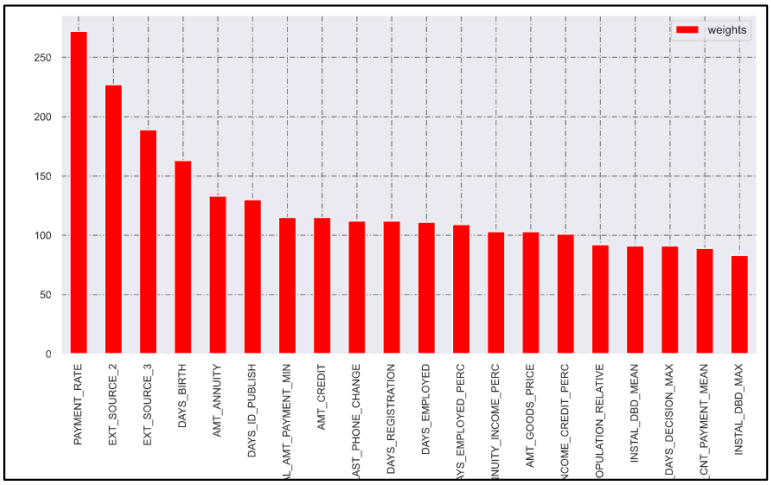


L'interprétabilité globale et locale du modèle

Afin de comprendre pourquoi un client donné est considéré comme bon ou mauvais prospect et quelles sont les données / Features qui expliquent son évaluation. Il est donc nécessaire à la fois, de connaître d'une manière générale les principales Features qui contribuent à l'élaboration du modèle, et de manière spécifique pour le client qu'elle est l'influence de chaque feature dans le calcul de son propre score (feature importance locale)

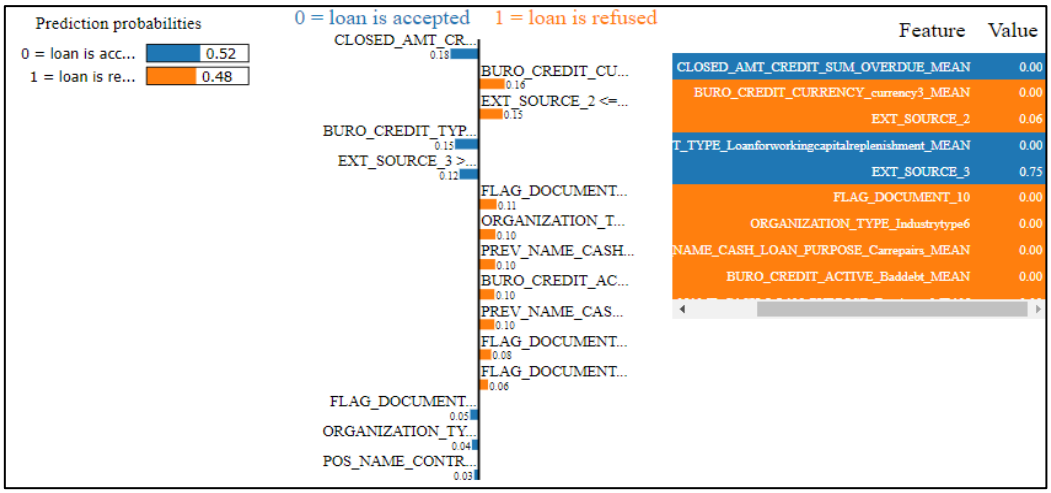
Concernant **feature importance globale** l'algorithme LGBMClassifier proposent cette fonctionnalité

```
coefficients = Scoring_credit.feature_importances_
```



Concernant Local Explanations LIME qui a été utilisé afin de fournir un calcul de feature importance locale indépendante de l'algorithme pour un client spécifique.

Local Interpretable Model-agnostic Explanations : est un modèle local qui cherche à expliquer la prédiction d'un individu par analyse de son voisinage.



On a également utilisé SHAP qui signifie SHapley Additive explanations et utilise l'approche de la théorie des jeux pour expliquer les prédictions du modèle.

Connue pour analyser les performances des modèles d'apprentissage automatique en interprétant les prédictions sur des exemples individuels, Les valeurs de Shapley calculent l'importance d'une variable en comparant ce qu'un modèle prédit avec et sans cette variable

Les limites et les améliorations possibles

- Le premier point c'est l'entrée du modèle, on parle bien du Data-Set donc, il faudrait discuter avec l'équipe de data engineer ou bien avec les personnes qui s'occupe de la collecte des données pour les exhorter à construire de base de données riches et complètes dans la limite possible, ainsi que se focaliser sur les Features pertinentes, il faudrait discuter également avec des experts du secteur des banquiers pour bien comprendre le besoin métier.

- De travailler plus sur la partie « feature engineering », la sélection de bonne Features de créer plus de variables pour enrichir le Data-Set, comprendre les relations entre variables et Targets et utiliser des méthodes de sélection de variables adapté comme par exemple **SelectKBest**

- Refaire l'entrainement avec une large liste d'algo, des méthodes de validation croisé, et une large liste d'hyperparamètres qui nécessite aussi du matériel professionnel puissants