



**FORMATION DATA SCIENTIST**

|  |  |
| --- | --- |
|  | **NOTE METHODOLOGOGIQUE** |

**Implémentez un modèle de scoring**

**Projet 7**

**Samir HINOJOSA DIAZGRANADOS**

Février 2022

Table des matières

[1. Introduction 3](#_Toc95129002)

[2. Présentation des donnes 3](#_Toc95129003)

[3. Modélisation 4](#_Toc95129004)

[3.1. Sélection du Kernel 4](#_Toc95129005)

[3.2. Optimisation du modèle 4](#_Toc95129006)

[3.2.1. Traitements initiaux 5](#_Toc95129012)

[3.2.2. Données déséquilibrées 5](#_Toc95129013)

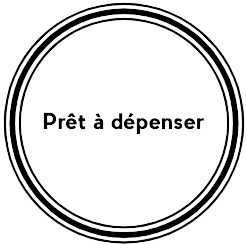
[3.2.3. Fonction coût 6](#_Toc95129014)

[3.2.4. Hyperopt 6](#_Toc95129015)

[4. Interprétation du modèle 7](#_Toc95129016)

[5. Limites et les améliorations possibles 7](#_Toc95129017)

# **Introduction**

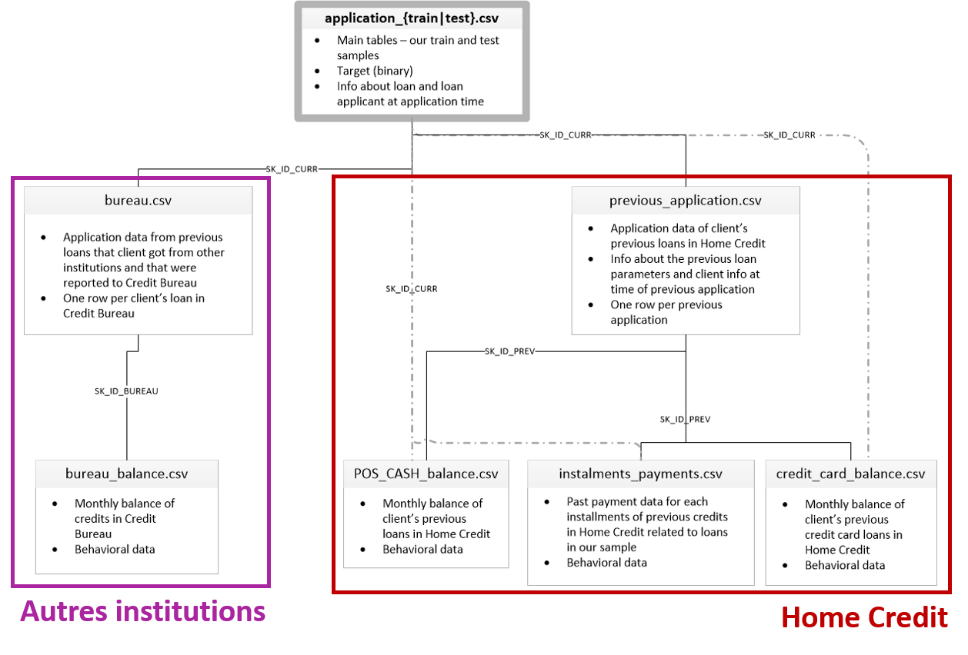
La société « **Prêt à dépenser** » souhaite mettre en œuvre un outil de « **scoring crédit** » pour calculer la probabilité qu’un client rembourse son crédit d’après diverses données.

La mission est développée un algorithme de classification qui classifie la demande en crédit accepté ou rejeté.

Prendre en compte :

* Sélectionner un kernel Kaggle pour faciliter la préparation des données nécessaires à l’élaboration du modèle de scoring.
* Déployer le modèle de scoring comme une API.
* Construire un tableau de bord interactif à destination des gestionnaires de la relation client permettant d'interpréter les prédictions faites par le modèle de scoring

# **Présentation des donnes**

****

Sept (7) tables de données sont mises à disposition pour la démarche.

Il y a de tables de données qui viennent des autres institutions comme :

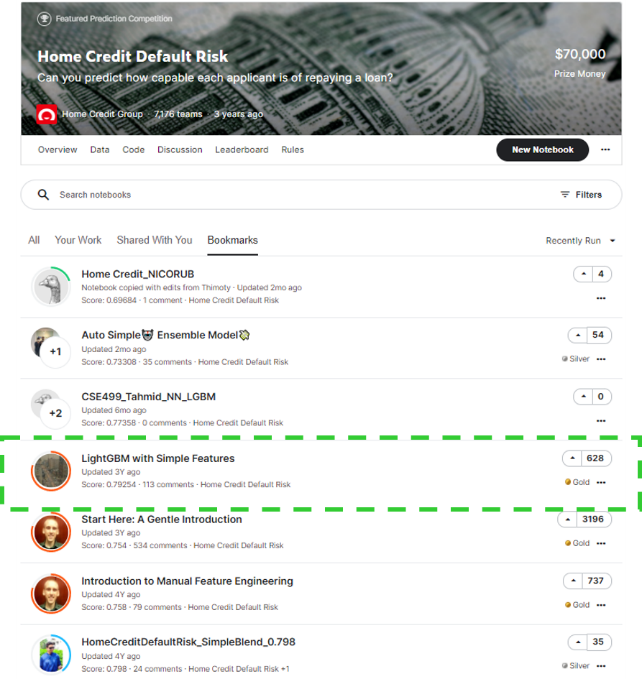
* Bureau : Antécénts de crédit du client (autres institutions financières)
* Bureau balance : Soldes mensuelles de crédits précédents

Aussi, on peut trouver de tables de données propre de « **Prêt à dépenser** »

* Previos\_application: Demandes de crédit antérieures
* Pos\_cash\_balance: Des bilans mensuels des anciens points de vente et des prêts cash
* Installements\_payments: Historique de remboursement des crédits précédemment
* Credit\_card\_balance: Solde mensuel des cartes de crédit antérieures

# **Modélisation**

## **Sélection du Kernel**

Le kernel utilisé pour le projet est

[LightGBM with Simple Features](https://www.kaggle.com/jsaguiar/lightgbm-with-simple-features).

La sélection est basée sur

* Un bon score dans la compétition
* Un Feature Engineering performant
* L’opportunité d'utiliser un Framework basé sur le Gradient Boosting
* Le kernel recommandé dans le projet

Tout au long du kernel, des traitements sont faits pour obtenir des nouvelles données

* Identification et traitement des variables catégorielles   
    
  Création de nouvelles variables

DAYS\_EMP\_% = DAYS\_EMP / DAYS\_BIRTH

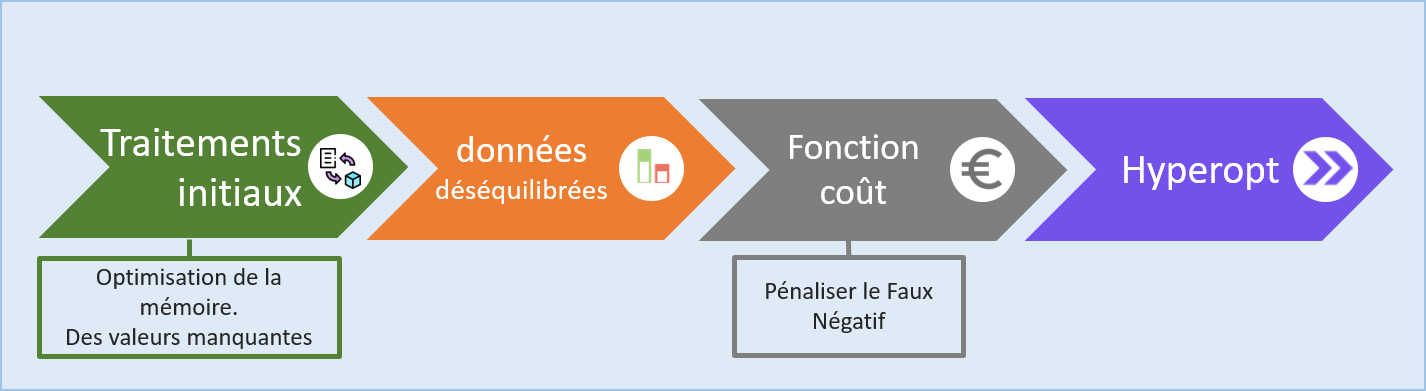
Min, Max, Mean, Sum, Var

* Unification de jeux des données

+700 variables en total

* Modélisation avec LGBMClassifier

## **Optimisation du modèle**



Pour faire l’optimisation du modèle, on va prendre en compte le Feature Engineering déjà fait dans le kernel choisi.



### **Traitements initiaux**

Tout d’abord, il faut faire une optimisation d’usage de la mémoire parce que l’utilisation actuelle est 2,1 GB. Alors, on va transformer les types de colonnes

* **if** df[col]**.**dtype **==** "int64" **and** df[col]**.**nunique() **==** 2:

df[col] **=** df[col]**.**astype("int8")

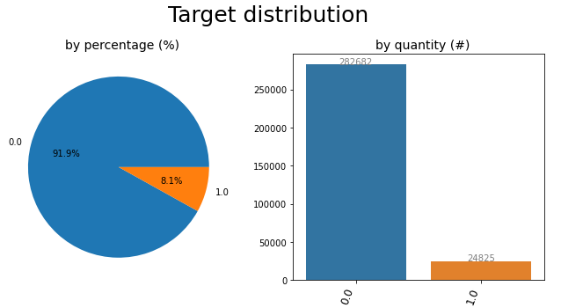
* **if** df[col]**.**dtype **==** "float64" **and** df[col]**.**min() **>=** **-**2147483648 **and** df[col]**.**max() **<=** 2147483648:

df[col] **=** df[col]**.**astype("float32")

À travers de cette transformation, l'usage de la mémoire est passé de 2.1Gb à 941 Mb.

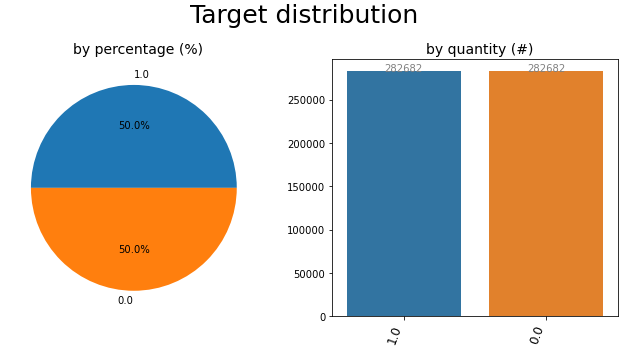
D’ailleurs, on a fait aussi, ***le traitement des valeurs manquantes*** à travers de l’imputation de la moyennée basée sur la colonne.

### **Données déséquilibrées**

L’analyse exploratoire a montré que la mission s’agit d’un problème de classification déséquilibré.

Une modification de l’ensemble de données est possible avant d’entraîner le modèle prédictif afin d’équilibrer les données (égaliser les clases).

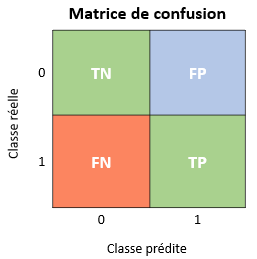
Dans ce cas, on va utiliser une stratégie appelée rééchantillonnage (re-sampling), faisons concentration sur la méthode le sur-échantillonnage (Oversampling) avec SMOTE[[1]](#footnote-1) (Synthetic Minority Oversampling Technique).



(0) Ce sont des prêts qui ont été remboursés et (1) Ce sont des prêts qui n'ont pas été remboursés.

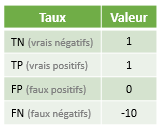
### **Fonction coût**

Pour ce projet, une fonction coût a été développée dans l’objectif de pénaliser des Faux Négatifs.



* **TN (Vrais Négatifs)** : des prêts qui ne sont pas en défaut et ont été prédits correctement
* **TP (Vrais Positifs) :** des prêts qui sont en défaut et ont été prédits correctement
* **FP (Faux Positif)** : des prêts qui ne sont pas en défaut et ont été prédites de manière incorrecte
* **FN (Faux Négatif)** : des prêts qui sont en défaut et ont été prédites de manière incorrecte

Un Faux Positif (FP) constitue une perte d'opportunité pour la banque, à la différence d'un Faux Négatif (FN) qui constitue une perte pour créance irrécouvrable.

  
Ces valeurs de coefficients signifient que les Faux Négatif (FN) engendrent des pertes 10 fois par rapport aux autres.

Ces poids ont été fixé arbitrairement et il est tout à fait envisageable de les modifier à la convenance de l’optique métier.

*# Total of default and not default cases*

total\_not\_default = TN + FP *# Not default cases*

total\_default = TP + FN *# Default cases*

*# Calculating the gains based on the rate/values*

gain\_total = TN\*TN\_rate + TP\*TP\_rate + FP\*FP\_rate + FN\*FN\_rate

gain\_maximun = total\_not\_default\*TN\_rate + total\_default\*TP\_rate

gain\_minumun = total\_not\_default\*TN\_rate + total\_default\*FN\_rate

*# Normalize to get score between 0 (baseline) and 1*

score = (gain\_total - gain\_minumun) / (gain\_maximun - gain\_minumun)

### **Hyperopt**

Pour réaliser l’optimisation du modèle, on a pris en compte l’outil HyperOpt, conçu pour automatiser la recherche d'une configuration optimale d'hyperparamètres basé sur une optimisation bayésienne et prise en charge par la méthodologie SMBO (optimisation globale de modèle séquentielle). Les hyperparamètres seront basés sur les paramètres déjà existants dans le kernel choisi

Il est nécessaire de dire que le kernel choisi a déjà un cross-validation mis en œuvre. C’est pour cela que dans cette partie, on va faire attention sur l'hyperparametrisation du modèle

D’ailleurs, pour optimiser la performance et le temps, on a pris en compte, les suivants points :

N\_ESTIMATORS **=** [8000, 10000, 12000]

NUM\_LEAVES **=** [32, 34, 36]

MAX\_DEPTH **=** [7, 8, 9]

space\_params **=** {

"n\_estimators": hp**.**choice("n\_estimators", N\_ESTIMATORS),

"learning\_rate": hp**.**uniform("learning\_rate", 0.002, 0.003),

"num\_leaves": hp**.**choice("num\_leaves", NUM\_LEAVES),

"max\_depth": hp**.**choice("max\_depth", MAX\_DEPTH),

"reg\_alpha": hp**.**uniform("reg\_alpha", 0.041545473, 0.051),

"reg\_lambda": hp**.**uniform("reg\_lambda", 0.0735294, 0.0835294),

"min\_split\_gain": hp**.**uniform("min\_split\_gain", 0.0222415, 0.0322415),

"min\_child\_weight": hp**.**uniform("min\_child\_weight", 39.3259775, 49)

}

* Mettre en œuvre le StandardScaler()
* Définir les suivants paramètres avant de lancer la modélisation (pour optimiser les temps d’exécution et pour le dire au modèle que le jeu de données est déjà équilibré.)

LGBMClassifier

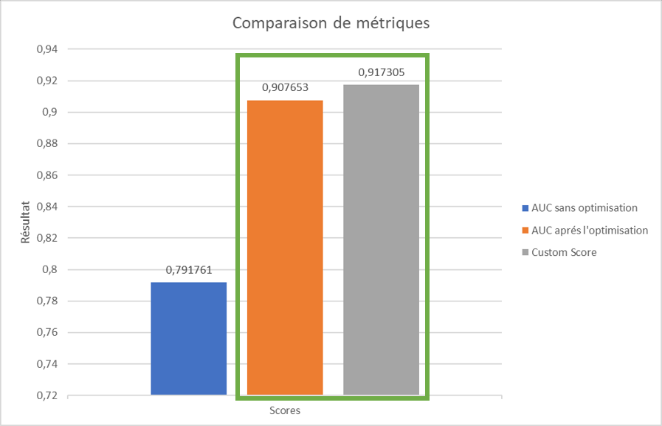
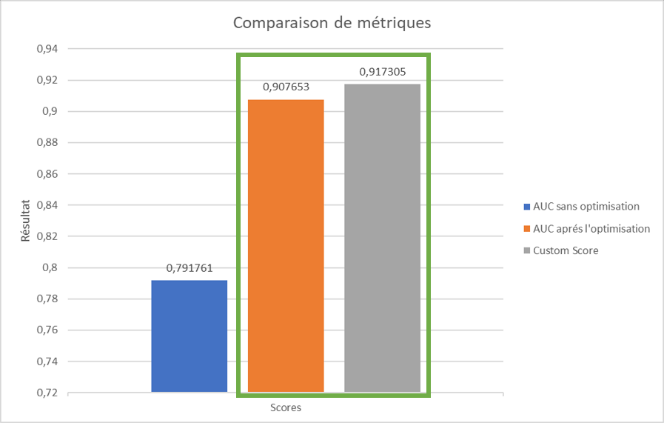
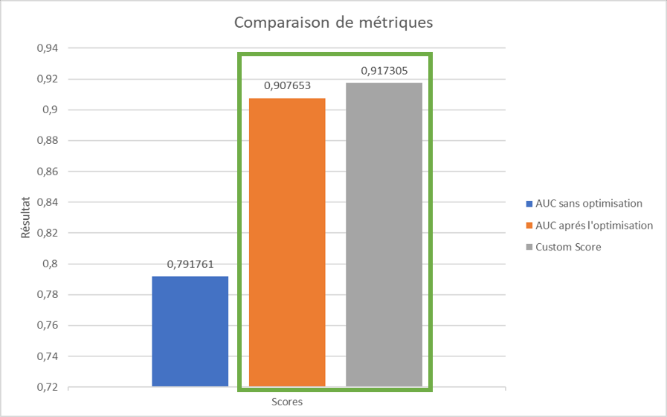
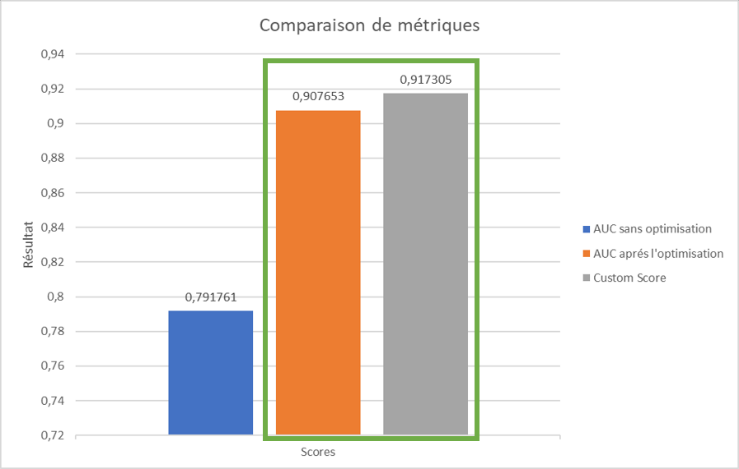
colsample\_bytree=0.8

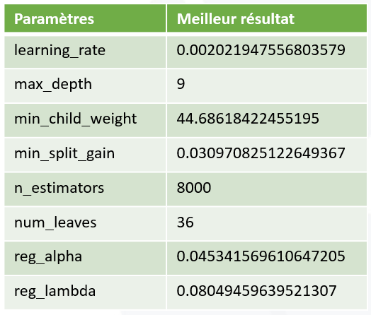
subsample=0.8

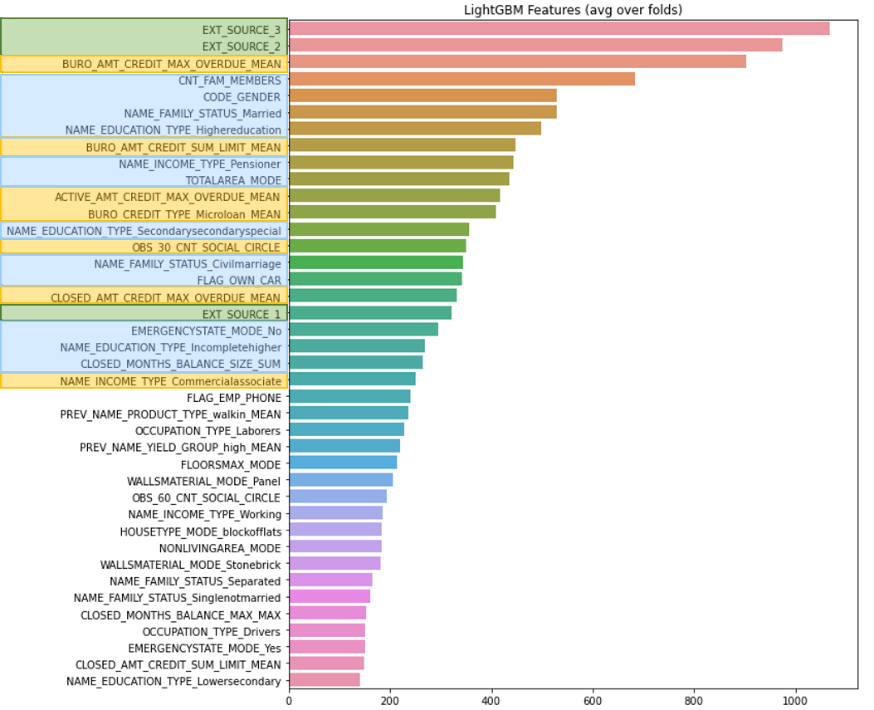
Is\_unbalance=False

# **Interprétation du modèle**

À travers l’optimisation du modèle fait, on a réussi des meilleurs scores que ceux du précédent. Voici les meilleurs paramètres du modèle





* Les variables les plus importantes proviennent d'une source **externe.** Ce sont des scores normalisés provenant d'autres institutions.
* Le Feature Engineering a ajouté la valeur au moment de la modélisation. On peut le remarquer à travers des terminaisons de noms de features (mean, max, etc.).
* Le modèle a pris en compte des différentes variables   
  ***Variables personnelles****,* ***Variables bancaires****,* ***Variables externes.***

# **Limites et les améliorations possibles**

* Il faut faire une analyse exploratoire au début pour bien comprendre les données.
* Il est nécessaire de faire une réduction de données pour éviter le « Curse of Dimensionality ». Il faut prendre en compte une méthode comme Feature Selection par Scikit-Learn
* Une analyse du composant principal « PCA » peut apporter des avantages
* Aller plus loin dans l'hyperparametrisation du modèle

1. SMOTE (Synthetic Minority Oversampling Technique) : Consiste à synthétiser des éléments pour la classe minoritaire, à partir de ceux qui existent déjà en choisissant aléatoirement un point de la classe minoritaire et à calculer les k plus proches voisins de ce point. [↑](#footnote-ref-1)