|  |
| --- |
| Mise en œuvre d’un outil de « scoring crédit »  **Il calcule la probabilité qu’un client rembourse son crédit, puis classifie la demande en crédit accordé ou refusé** |

|  |
| --- |
| **DESCRIPTION**  Ce document décrit la mise en œuvre de cet outil de « scoring crédit », de la gestion des données à la construction du modèle en passant par l’amélioration et l’analyse du modèle. |

Table des matières

[INTRODUCTION 3](#_Toc125702705)

[LES DONNEES 3](#_Toc125702706)

[METHODOLOGIE D’ENTRAÎNEMENT DU MODELE 5](#_Toc125702707)

[1. FEATURE ENGINEERING 5](#_Toc125702708)

[2. les donnees 6](#_Toc125702709)

[3. Prendre les meilleurs features 6](#_Toc125702710)

[4. REEQUILIBRAGE DES DONNEES 6](#_Toc125702711)

[5. entraÎNEMENT DU MODELE 6](#_Toc125702712)

[6. amelioration DU MODELE 6](#_Toc125702713)

[LA FONCTION COÛT METIER, L’ALGORITHME D’OPTIMISATION, LA METRIQUE D’EVALUATION 7](#_Toc125702714)

[1. LA FONCTION COÛT METIER 7](#_Toc125702715)

[2. L’ALGORITHME D’OPTIMISATION 7](#_Toc125702716)

[3. metriques d’EVALUATION 7](#_Toc125702717)

[ACCURACY, PRECISION, RECALL, F1 SCORE , FBETASCORE 7](#_Toc125702718)

[L’INTERPRETABILITE GLOBALE ET LOCALE DU MODELE 8](#_Toc125702719)

[1. ADABOOST CLASSIFIER AVEC SMOTE ET FBETASCORE 8](#_Toc125702720)

[ACCURACY, PRECISION, RECALL, FBETA SCORE 8](#_Toc125702721)

[LA MATRICE DE CONFUSION 8](#_Toc125702722)

[COURBE ROC ET SCORE AUC 8](#_Toc125702723)

[les limites et les ameliorations possibles 9](#_Toc125702724)

# INTRODUCTION

Nous sommes Data Scientist au sein d'une société financière, nommée **"Prêt à dépenser"**, qui 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” pour calculer la probabilité**qu’un client rembourse son crédit, puis classifie la demande en crédit accordé ou refusé. Elle souhaite donc développer un **algorithme de classification** en s’appuyant sur des sources de données variées (données comportementales, données provenant d'autres institutions financières, etc.).

De plus, les chargés de relation client ont fait remonter le fait que les clients sont de plus en plus demandeurs de **transparence** vis-à-vis des décisions d’octroi de crédit. Cette demande de transparence des clients va tout à fait dans le sens des valeurs que l’entreprise veut incarner.

**Prêt à dépenser**décide donc de **développer un dashboard interactif** pour que les chargés de relation client puissent à la fois expliquer de façon la plus transparente possible les décisions d’octroi de crédit, mais également permettre à leurs clients de disposer de leurs informations personnelles et de les explorer facilement.

# LES DONNEES

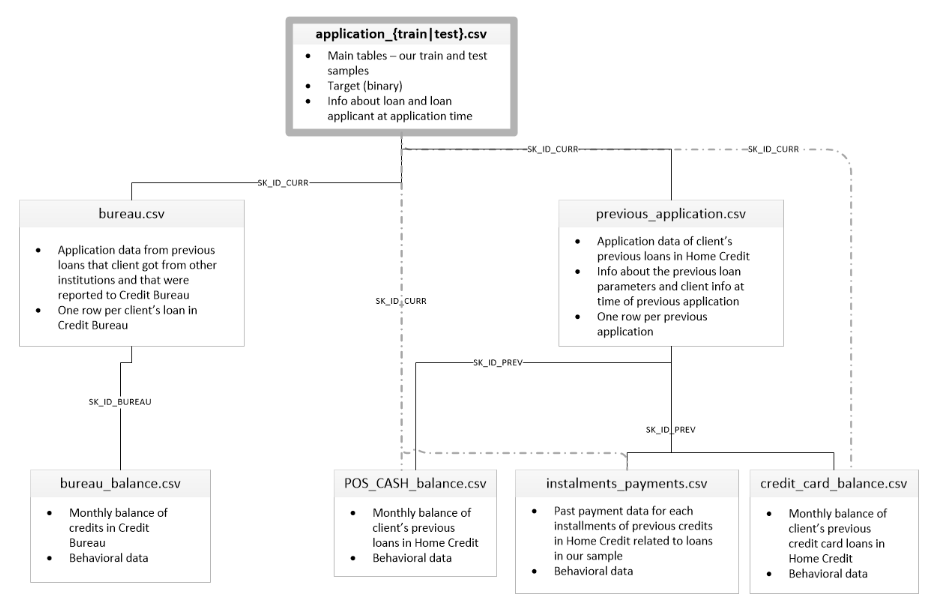
Les données se trouvent à l’adresse : <https://www.kaggle.com/c/home-credit-default-risk/data>

Nous disposons de 9 fichiers CSV : **application\_{train|test}.csv, bureau.csv, bureau\_balance.csv, POS\_CASH\_balance.csv, credit\_card\_balance.csv, previous\_application.csv, installments\_payments.csv, HomeCredit\_columns\_description.csv**

Dans le fichier application\_train.csv, **TARGET**, est la "Variable cible (1 - client ayant des difficultés de paiement : il/elle a eu un retard de paiement de plus de X jours sur au moins un des Y premiers versements du prêt dans notre échantillon, 0 - tous les autres cas)"

On peut voir ci-dessous les liens entre les différents fichiers.

*Nous avons aussi à disposition une base de données « test ». Cette ne nous a pas été utile à la construction et à l’entraînement du modèle. Son utilisation a été extérieure au sujet de la présente note technique.*



[2] Lien Kernel Kaggle : <https://www.kaggle.com/willkoehrsen/start-here-a-gentle-introduction>

[4] Librairie Python « Imblearn » pour équilibrage des données : <https://imbalanced-learn.readthedocs.io/en/stable/user_guide.html>

# METHODOLOGIE D’ENTRAÎNEMENT DU MODELE

Il était mis à notre disposition un kernel kaggle pour le feature engineering.

## FEATURE ENGINEERING

## les donnees

Nous faisons la séparation entre les données d’entrainements et les données test avec 70% de données d’entrainements et 30% de test.

## Prendre les meilleurs features

On prend les meilleurs features.

## REEQUILIBRAGE DES DONNEES

Les données sont déséquilibrées. Il existe des algorithmes pour **générer des échantillons synthétiques de manière automatique**. Le plus populaire de ces algorithmes est **SMOTE**(pour Synthetic Minority Over-sampling Technique). Comme son nom l’indique, SMOTE est une **méthode de sur-échantillonnage**. Elle fonctionne en créant des échantillons synthétiques à partir de la classe minoritaire au lieu de créer de simples copies. Source : [https://datascientest.com/]

## entraÎNEMENT DU MODELE

On entraine le modèle avec les données déséquilibrées et les données rééquilibrées.

## amelioration DU MODELE

On cherche les meilleurs hyperparamètres avec RandomizedSearchCV

# LA FONCTION COÛT METIER, L’ALGORITHME D’OPTIMISATION, LA METRIQUE D’EVALUATION

## LA FONCTION COÛT METIER

On va utiliser les hypothèses suivantes pour chaque candidat au crédit :

1. frais généraux pour chaque client : coût = 1.
2. gain par client qui rembourse le crédit = 10 (les "True Negative")(octroi de crédit à un client solvable).
3. coût par client qui fait défaut = 100 (les "False Negative)(octroi de crédit à un client non-solvable).
4. absence de coûts additionnels pour les "True Positive" (refus de crédit à un client non-solvable).
5. absence de coûts additionnels pour les "False Positive" (refus de crédit à un client solvable).

Une image contenant texte

Description générée automatiquement

Plutôt qu'utiliser une fonction coût, on va aussi travailler avec une fonction gain construite ainsi :

Une image contenant texte

Description générée automatiquement

## L’ALGORITHME D’OPTIMISATION

**RandomizedSearchCV** implémente une méthode "fit" et une méthode "score".  Contrairement à **GridSearchCV,** toutes les valeurs de paramètre ne sont pas testées, mais un nombre fixe de paramètres est échantillonné à partir des distributions spécifiées. Le nombre de paramétrages essayés est donné par n\_iter.

Si tous les paramètres sont présentés sous forme de liste, un échantillonnage sans remise est effectué. Si au moins un paramètre est donné sous forme de distribution, l'échantillonnage avec remise est utilisé. Source : [https://scikit-learn.org/]

## metriques d’EVALUATION

### ACCURACY, PRECISION, RECALL, F1 SCORE , FBETASCORE

* L’accuracy détermine la justesse de la prédiction.
* La Précision détermine que, quand le classifieur déclare que la prédiction est un 1, il a raison à z%.
* Le Recall : Ce coefficient détermine le pourcentage de détection des 1 du classifieur.
* F1 Score :
* Fbetascore :
* La courbe ROC : l’aire sous la courbe pour un classifieur parfait a un score AUC égal à 1, alors qu’un classifieur aléatoire a un score AUC de 0.5.

# L’INTERPRETABILITE GLOBALE ET LOCALE DU MODELE

## ADABOOST CLASSIFIER AVEC SMOTE ET FBETASCORE

Après étude de plusieurs modèles de classification, notre choix s’est porté sur le Adaboost Classifier avec smote et fbetascore.

### ACCURACY, PRECISION, RECALL, FBETA SCORE

\*\*\*\*\*Mesures de performance avec beta = 10\*\*\*\*\*\*\*\*

Accuracy = 0.8061333333333334

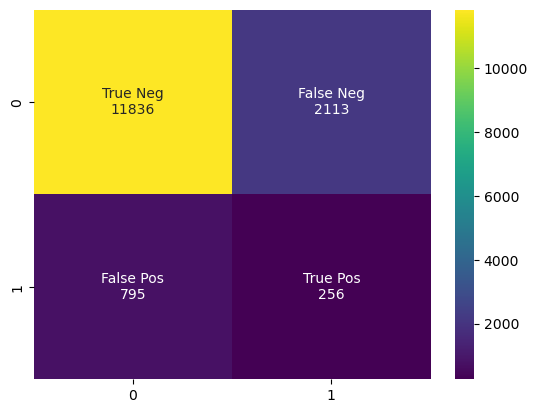
Precision = 0.5225610444249624

Recall = 0.5460485761676745

Fbeta\_Score = 0.24059030976374585

### LA MATRICE DE CONFUSION

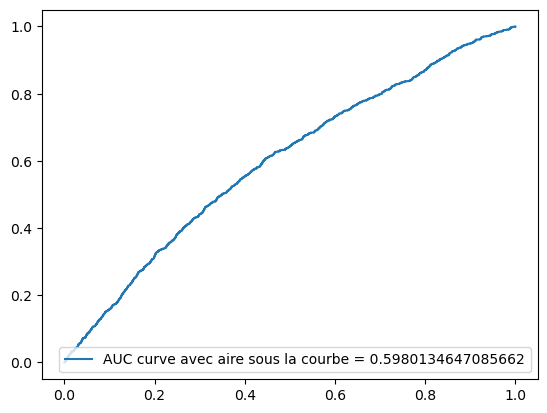
**PRINCIPE :** La matrice de confusion aide à le nombre de fois où des clients de la classe 0 ont été rangées dans la classe 1. Si par exemple nous voulons connaître le nombre de fois où le modèle de classification à bien réussi à classer une classe 0, on examinera la cellule des True Neg  qui est à l’intersection de la ligne et de la colonne 0.



Notre modèle idéal serait de retrouver 100% des True Positif, ce sont en fait les clients qui ne remboursent pas leur prêt. C’est aussi la catégorie la plus difficile à prédire car minoritaire dans l’échantillon.

### COURBE ROC ET SCORE AUC

**PRINCIPE :** La courbe ROC (Receiver Operating Characteristic) Nous pouvons comparer les modèles de classification à l’aide de l’aire sous la courbe (Area Under the Curve ou AUC). Un modèle parfait aurait un score AUC égal à 1, alors qu’un modèle aléatoire aurait un score AUC de 0.5.



# les limites et les ameliorations possibles

Nous avons pu constater tout au long de ce document que les performances du modèle ne sont pas bonnes. Pour résumé : Au mieux, notre modèle peut trouver 55% des classes 1(recall), et lorsqu’il en prédit une, il a raison à 53%.

Nous pouvons l’expliquer par le features engineering qui est à améliorer. En effet, le Kernel choisi est plutôt pauvre sur le traitement des données. Il ne crée pas beaucoup de variable qui peuvent être utiles à un modèle de classification comme des moyennes, des médianes, des écarts-types, et ça pour plusieurs features. Il existe peut-être un Kernel plus abouti qui permettra une meilleure performance prédictive au modèle. Sinon, prendre le temps de réaliser nous même notre feature engineering, ce qui nous permettra de bien comprendre nos données et ainsi construire un feature engineering adapté à notre besoin.