Note méthodologique

I. La méthodologie d'entraînement du modèle :

Avant de parler de la méthodologie d'entraînement du modèle, parlons du jeu de données qui va nous permettre d'entraîner les modèles. En effet, le jeu de données est composé de plus de 300 000 clients de plus de 700 variables qui permettent d'avoir des informations sur les différents clients.

1. Prétraitement du jeu de données :

Il faut tout d'abord supprimer, arbitrairement, les colonnes avec plus de 50 % de valeurs manquantes. Ainsi, on se retrouve avec un jeu de données avec un peu plus de 500 variables.
Avec ce nouveau jeu de données, il a fallu faire un traitement des valeurs aberrantes en supprimant 0.5 % des valeurs extrêmes pour chaque variable. Dans cette partie, l'utilisation du z-score a permis, après avoir centré et réduit les variables, de supprimer les valeurs supérieurs à 3 écart types de la moyenne. Dans ce cas les valeurs de z suivent une loi normale centré réduite car nous avons un peu plus de 300 000 clients ce qui est équivalent a une limite de à l'infinie dans notre cas.
De plus, j'ai fait un traitement des valeurs manquantes en imputant les valeurs manquantes par la médiane. Ici, il aurait pu être plus judicieux d'utiliser IterativeImputer (pour les variables corrélées entre elles) et KNN (pour le reste des variables) mais avec plus de 500 variables le code mettait beaucoup plus de temps.
Puis, une ADE nous donne une meilleure compréhension du jeu de données. En effet, en faisant des graphiques pour les variables qui sont les plus importantes d'après le kernel, on peux voir s'il y a une concordance entre la variable à expliquer avec les autres variables.

2. Surreprésentation d'accordance de crédit :

Le jeu de données a une particularité. En effet, la variable à expliquer a une surreprésentation de 0. Cette surreprésentation de 0 doit être prise en compte dans les modèles sinon les modèles feront que de prédire 0 ce qui n'est pas une bonne solution. Ce qui veut dire qu'on va donner des crédits à tout le monde. Pour pallier cette surreprésentation de 0, une méthode est d'utiliser un oversampling de type smote.

3. Sélection du modèle

Pour l'entraînement du modèle, il est nécessaire d'utiliser plusieurs modèles de classification. Les modèles de classification utilisés sont les suivants : la régression logistique, l'arbre de décision, la forêt aléatoire et enfin le gradient boosting. Pour ces 4

modèles, j'ai utilisé la classe gridsearchCV pour choisir les meilleurs hyperparamètres de chaque modèle. GridsearchCV permet de faire de la validation croisée en comparant un modèle avec différents paramètres entre eux en fonction d'une métrique. Finalement, le Random Forest donnait le meilleur résultat avec la fonction de coût métier.

Modèle	score
Régression logistique	-0.04983618285732868
Arbre de décision	0.11201352812533028
Forêt aléatoire	0.14886628781411834
Gboost	

4. Démarche de modélisation

La première étape est d'utiliser la fonction smote pour équilibrer le nombre de clients qui ne peuvent pas avoir de crédit. En effet, le jeu de données comporte plus de 90 % de clients qui peuvent avoir un crédit contre seulement 10 %.
Puis il faut utiliser la classe gridsearchCV pour sélectionner les hyperparamètres optimaux pour chaque modèle par rapport à la fonction de coût métier. La fonction va donner le meilleur score ainsi que les hyperparamètres pour chaque modèle ce qui va permettre de choisir le meilleur parmi tous les modèles.
Enfin, une interprétabilité du modèle sera faite en faisant une feature importance locale et globale à l'aide de deux librairies lime et shap.

II. La fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation :

Un modèle est dit meilleur qu'un autre lorsqu'il fournit de meilleurs résultats par rapport à une certaine métrique. 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.
La procédure est la suivante :
Il s'agit d'un problème de classification binaire. Donc, il existe donc 4 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 1
- FN : False Negatif, le modèle prédit 0 alors que la valeur réelle est de 1

- FP : False Positif, le modèle prédit 1 alors que la valeur réelle est de 0

Par conséquent, le modèle peut se tromper de deux manières différentes, en prédisant un 1 lorsque c'est un 0 (faux positif) et en prédisant un 0 lorsque c'est un 1 (faux négatif). En revanche, la perte d'argent est plus conséquente pour un FN (prêt accordé alors que le client n'a pas les moyens de payer) qu'un manque à gagner pour un FP (prêt non accordé alors que le client a les moyens de payer). Une fonction coût a donc été créée pour tenir compte de l'importance relative de chaque erreur qui est la suivante :

$$\frac{(TN + TP) - (10{*}FN + FP)}{taille\ de\ l'échantillon}$$

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

III. L'interprétabilité globale et locale du modèle :

L'interprétabilité d'un modèle permet de mettre en avant les variables qui influencent la prédiction. Le modèle sélectionné étant assez complexe,deux librairies ont été utilisées. La librairie shap pour la feature importance globale et la librairie lime pour la feature importance locale.

1. Interprétation globale:

A l'aide de la librairie shap, on peut voir plusieurs variables qui influence la prédiction, ce sont les variables suivantes :

- EXT_SOURCE_2 : un score à partir d'une source de données externe;
- CODE_GENDER : le sexe du clients;
- HOUSETYPE_MODE_block of flats : si le client habite dans un appartement;
- EMERGENCYSTATE_MODE_No : si le client n'est pas en état d'urgence;
- NAME_FAMILY_STATUS_Married : si le client est marié;
- FLAG_OWN_CAR : si le client possède une voiture;
- WALLSMATERIAL_MODE_Panel : si le client possède des murs avec des panneau dans son logement;
- FONDKAPREMONT_MODE_reg oper account :
- ORGANIZATION_TYPE_XNA : si le client travaille dans une organisation de type XNA;
- NAME_INCOME_TYPE_Pensioner : si le client est retraité.

2. Interprétation local :

Grâce à la librairie lime, on peut connaître les variables qui ont influencé la prédiction pour un seul client. Ici les variables qui ont influencé la prédiction du client 207965 :

- NAME_EDUCATION_TYPE_Secondary/secondary spécial = 1
- WALLSMAERIAL_MODE_Panel = 0
- FLAG_OWN_CAR = 0
- ORGANIZATION_TYPE_Transport:type 1 = 0
- HOUSETYPE_MODE_block of flats = 1

Ici le client est solvable mais les quatres premières variables influencent la prédiction pour que le client n'est pas solvable tandis que la dernière montre le que le client est solvable (voir le notebook). Ainsi cela montre que pour être solvable les client doivent avoir un niveau d'étude supérieur au secondaire, ne pas avoir de mur de type panneau, doit posséder une voiture, ne devrait pas travailler dans une organisation de type transport de niveau 1 et peux vivre dans une appartement.

IV. Les limites et les améliorations possibles :

1. Prétraitement :

Le prétraitement du jeu de données aurait été meilleur si le jeu de données était plus petit. En effet, dans la partie imputation des données, il aurait été plus pratique de faire une imputation par IterativeImputer pour les variables corrélées entre elles et une imputation par KNN pour le reste des variables. Un imputation par la médiane n'est sûrement pas ici la meilleure solution car les variables ne sont certainement pas homogènes.

2. Sélection du modèle :

Dans la partie sélection du modèle, il a fallu utiliser un échantillon du jeu de données car la machine plante. Ainsi, l'échantillon a été choisi de manière aléatoire avec une stratification par la target mais est-ce qu'elle est représentative de la population.
De plus, on aurait pu utiliser d'autres modèles tels que les modèles de machines learning.

3. Interprétabilité du modèle :

Dans la partie interprétabilité du modèle, les variables ne sont pas très bien décrites dans le dictionaire de données **HomeCredit_columns_description** surtout pour les variables EXT_SOURCE_2, WALLSMAERIAL_MODE_Panel et EMERGENCYSTATE_MODE_No. Ici on peut seulement faire des suppositions d'après le nom de la variable.