NOTE MÉTHODOLOGIQUE DU PROJET 7

Version avant 14/12/2022

Intitulé du Projet : Implémenter un Scoring Client

Stagiaire : Landry Didier GAMASSA

OpenClassrooms : Parcours DATA SCIENTIST

Soutenance : Janvier 2023

Table des Matières

1 Contexte 2
2 Méthodologie d'entrainement du modèle 2
3 Fonction cout, algorithme d'optimisation et métrique d'évaluation 4
3.1 Fonctions cout 4
3.2 Métriques d'évaluation 4
4 L'interprétabilité globale et locale du modèle 6
5 Les limites et les améliorations possibles 7
5.1 Les limites : 7
5.2 Les améliorations : 7

1 Contexte

Ce projet consiste à développer au profit d'une société de crédit à la consommation dénommée << Prêt à dépenser>>, un modèle de scoring de la probabilité de défaut de paiement d'un client avec pas ou peu d'historique de prêt.

Les données utilisées pour ce projet sont une base de données de 307511 clients comportant 230 variables et repartie sur 8 tables des données dont 1 table principale déclinée en deux jeux de données : Application Train et Application Test.

Ce mémoire fait partie des livrables du projet et décrit les informations complémentaires suivantes :

- La méthodologie d'entraînement du modèle
- La fonction coût métier, l'algorithme d'optimisation et la métrique d'évaluation
- L'interprétabilité globale et locale du modèle
- Les limites et les améliorations possibles

2 Méthodologie d'entrainement du modèle

En machine Learning, il est fréquent d'entrainer des modèles sur des données présentant un déséquilibre des classes et tell est le cas de notre étude.
Après préprocessing et agrégation des données, le jeu des données agrégés a été entrainé sur un ensemble d'apprentissage (Xtrain, y_train), puis examiné à l'aide d'un ensemble de test (Xtest, y_test)
Au cours de notre analyse exploratoire, nous avons détecté que notre cas d'étude présentait une classification binaire avec des classe de clients composés d''une part de 92% des clients avec demande de prêts garantis et d'autre part 8% des clients avec demande de prêts non garantis.

Les méthodes de machine Learning classique ne sont pas toujours adaptées sur la classification des données déséquilibrés, elles donnent souvent des mauvais résultats et pire encore elles peuvent induire en erreur avec des scores trop optimistes

En classification binaire, les deux méthodes fréquemment utilisées pour améliorer les performances des modèles sont :

A. **La méthode data-Levels solution** qui consistent en des transformations opérées sur les données numériques en entrée d'un modèle de Machine Learning pour atténuer le déséquilibre (undersampling et oversampling). Dans le cas de notre projet, nous avons utlisé **SMOTE** qui est une technique de suréchantillonnage avec plusieurs variantes. Plutôt que de réduire la taille de la classe majoritaire, il permet d'agrandir la taille de la classe minoritaire.

B. **La méthode ensemble Learning** qui se reposent sur des adaptations des modèles de machine Learning classiques afin qu'il soit en mesure de mieux gérer une classification déséquilibrée.

Le jeu de données post feature-engineering a été exporté et utilisé sur la base d'un échantillon de 80% du jeu de données initial. Le but étant de comparer, avec un temps de calcul réduit, les approches de type Données équilibrée par SMOTE et type Données agrégées déséquilibrées.

Nous avons commencé par sélectionner l'algorithme Logistique Régression comme modèle de base.

Pour rappel notre étude va porter sur la probabilité de risque du non remboursement d'un prêt par un client(valeur=1)

L'algorithme de régression logistique propose de tester un modèle de régression dont la variable dépendante est dichotomique (codée 0-1) et dont les variables indépendantes peuvent être continues ou catégorielles.IL permet aussi de prédire la probabilité qu'un évènement arrive (valeur de 1) ou non (valeur 0) à partir de l'optimisation des coefficients de régression.

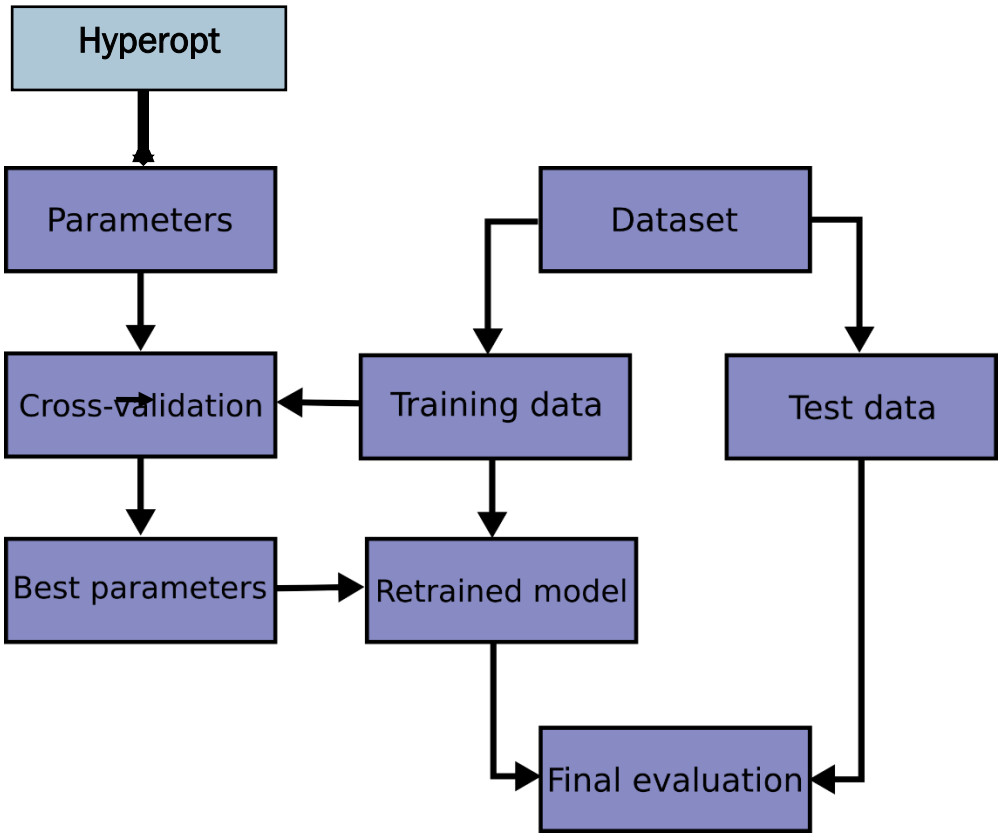
Dans notre étude, pour chaque approche (équilibrée et déséquilibré) les performances comparées ont permis de déterminé un modèle unique Baseline avec un Score_Roc_Auc=0.56 soit un modèle doté d'un pouvoir prédictif médiocre et très proche du jeter de pièce.

Le jeu de données post feature-engineering a été exporté et utilisé sur la base d'un échantillon de 80% de du jeu de données initial. Le but étant de comparer, avec un temps de calcul réduit, les approches de type données équilibrées et type données agrégées déséquilibrées.

Ensuite, nous avons sélectionné 4 algorithmes ensemblistes adaptés à la classification des données déséquilibrées. Il s'agit des algorithmes ensemblistes suivant : **AdaBoostClassifier, LGBMClassifier, XGBClassifier et RandomForestClassifier**

Pour chaque approche, tous ces modèles ont été testés et nous en avons extrait le meilleur Prédicteur.

Le choix du meilleur modèle a été effectué en retenant le modèle avec le meilleur score sur le jeu de validation.
Notre évaluation nous a permis de sélectionner l'Algorithme LightGBMClassifier et le oversampling de la classe minoritaire n'a pas apporté des meilleures performances.
Une fois le meilleur prédicteur sélectionné, nous avons alors procédé à la sélection des variables intégrées à sa construction par un algorithme RFE (Récursive Features Evaluation).
La sélection des variables pertinentes par RFE, nous a permis d'obtenir un jeu de données réduit.
Ensuite nous avons optimisé les performances du meilleur Prédicteur par la recherche et sélection de ses hyperparamètres selon l'algorithme GridSearchcv combiné si possible à Stratified Cross-validation
Enfin construction du modèle optimisé et entrainement sur le jeu de données réduit.
Synthétisons tout ce qui précède par le workflow ci-dessous :



<u>Observation : Le</u> fine tuning des hyperparamètres du meilleur modèle sélectionné pouvait se faire par GridSearchCv, par RamdomSearchCv mais l'algorithme Hyperopt par sa rapidité nous a semblé le mieux adapté.

3 Fonction cout, algorithme d'optimisation et métrique d'évaluation

3.1 Fonctions cout

Il s'agit d'une fonction qui détermine les performances d'un modèle d'apprentissage automatique pour un ensemble de données donné.

Les fonctions cout pour les algorithmes entrainés sont les suivantes :

Algorithme de Classification Binaire	Algorithm Fonction de Cout
AdaBoost Classifier (Decision Tree Algorithm)	• Minimisation de l'entropie de shanon à chaque noeud : Entropy = - $(p_1 \times \log_2(p_1) + p_2 * \log_2(p_2))$ ou • Minimisation de l'indice de Gini à chaque noeud : Gini = $1 - (p_1)^2 - (p_2)^2$
LGBM Classifier (Ensemble model Decision Tree Algorithm)	• Gradient-based One-Side Sampling
XGBoost Classifier (Ensemble model Decision Tree Algorithm	• Optimisation du Gain à chaque nœud par rapport à gamma.
RandomForest Classifier (Decision Tree Algorithm	• Minimisation de l'Entropie à chaque noeud ou • Minimisation de l'Indice de Gini à chaque noeud
Logistic-Regression	• Gradient descendant

3.2 Métriques d'évaluation

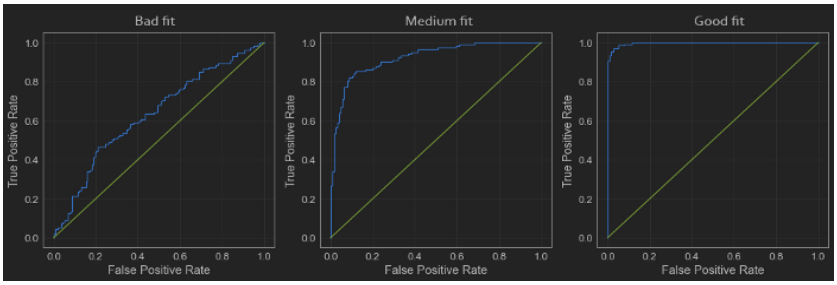
En machines Learning, entrainer un modèle sur des données déséquilibrée exige de sélectionner les métriques d'évaluation qui ne seront pas affectées par la mauvaise répartition des classes selon la démarche suivante :

a. **Construire une matrice de confusion du modèle.**
b. **Utiliser les métriques qui seront beaucoup moins influencées par la classe majoritaire telles que : Roc_Auc_Score, Le Recall(sensibilité) et le F_Score.**
c. **Interpréter la courbe ROC (Receiver Operating Characteristic) qui est un moyen très efficace d'évaluer la puissance de prédiction d'un classificateur.**

La courbe ci-dessous nous montre le comportement du classificateur pour chaque seuil en traçant deux variables : le taux des vrais positifs (**TPR**) et le taux de faux positifs (**FPR**).

Matrice de confusion

		Classe réelle	
		-	+
Classe prédite	-	True Negatives *(vrais négatifs)*	False Negatives *(faux négatifs)*
	+	False Positives *(faux positifs)*	True Positives *(vrais positifs)*



Sur les graphiques ci-dessus, la ligne verte représente où TPR=FPR, tandis que la ligne bleue représente la courbe ROC du classificateur. Si elle est exactement sur la ligne verte, cela signifie que le classificateur a le même pouvoir prédictif que lancer une pièce. Sur le graphique de gauche, la ligne bleue est relativement proche de la verte, ce qui signifie que le classificateur est mauvais.

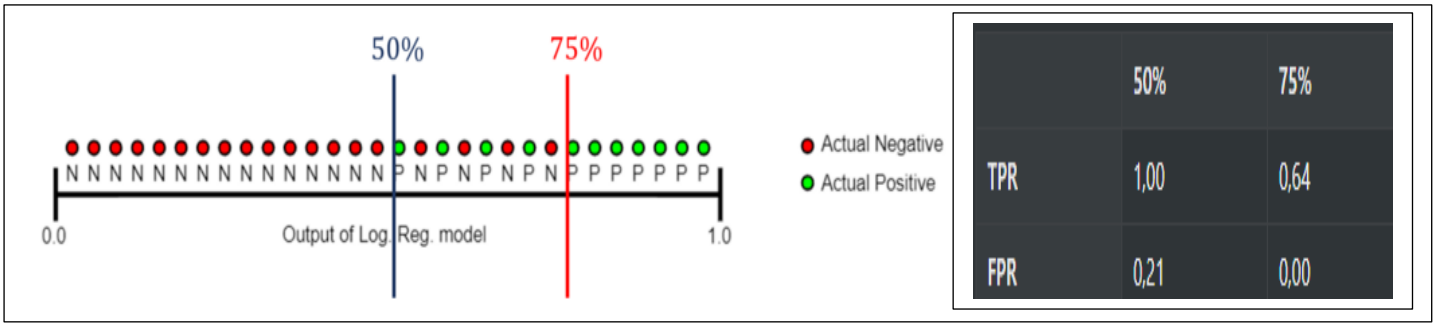
Le graphique le plus à droite montre un bon classificateur, avec la courbe ROC plus proche des axes et le « coude » proche de la coordonnée (0,1). Celui du milieu est un classificateur assez bon, plus proche de ce qu'il est possible d'obtenir à partir de données du monde réel.
Ci-dessous les formules utiles pour l'évaluation des qualités prédictives d'un classificateur binaire.

$$TPR=Recall=(sensitivity) = \frac{vrais\ positifs}{vrais\ positif + faux\ négatifs}$$

$$FPR= (1 - Specifity) = \frac{faux\ positifs}{vrais\ positifs + faux\ positifs}$$

$$Précision=\frac{vrais\ positifs}{vrais\ positifs + faux\ positifs}$$

Sur l'image ci-dessous, nous illustrons la sortie d'un modèle de régression logistique pour un dataset. Quand, nous fixons le seuil à 50%, aucune observations positive ne sera classée comme négative, donc FN=0 et TP=11, mais 4 observations négatives seront classées comme positive, donc FP=4, et 15 observations négatives seront classées comme négative, donc TN=15.



	50%	75%
TPR	1,00	0,64
FPR	0,21	0,00

Nous pouvons calculer le TPR et FPR pour chaque seuil et les comparer. En effet, en déplaçant le seuil à 75%, il apparait à sa droite seulement 7observations qui sont classées comme positive ainsi **TP=7** et **FP=0**. A gauche de ce seuil, Il apparait que toutes les observations négatives sont classées négatives donc **TN=19** et quatre observations positives sont classées négatives, donc **FN=4.**
Il en résulte que la sélection du seuil est tributaire des objectifs du modèle. Observons sur notre exemple, qu'il demeure plus important d'avoir toutes les observations positives classées comme positive en dépit de quelques observations négatives classées comme positive, le seuil fixé à 50% s'avère être le meilleur seuil.
Du point de vue d'une banque, le Recall sera plus important que la précision car on préfèrera limiter un risque de perte financière plutôt qu'un risque de perte de client potentiel. Par conséquent, on va donc faire usage d'une fonction qui optimisera, le cas échéant, les deux critères en donnant plus d'importance au Recall.Selectionner un seuil adéquat reviendra à sélectionner un coefficient Beta adéquat dans la fonction Fbeta_score :

$$0 \leq F_\beta = (1 + \beta^2) * \frac{precision * recall}{\beta^2 * precision + recall} \leq 1$$

.

Alors, notre objectif métier sera tributaire du coefficient β selon le tableau de variation ci-dessous :

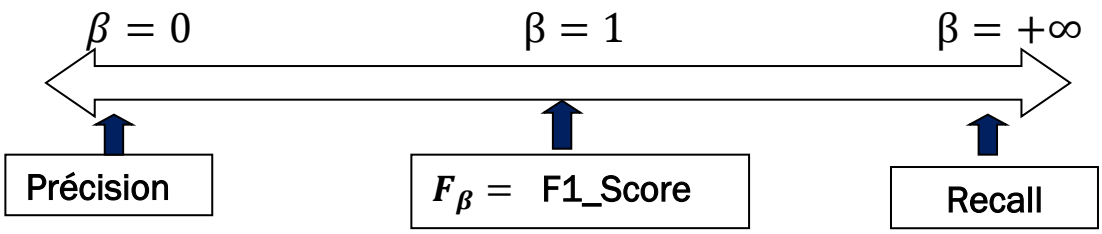
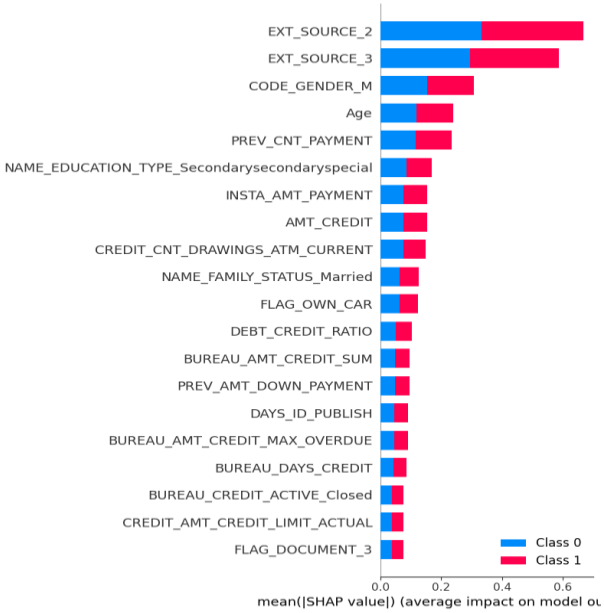


Illustration cas Scoring Client : Voir Notebook

4 L'interprétabilité globale et locale du modèle

- Interprétabilité globale :

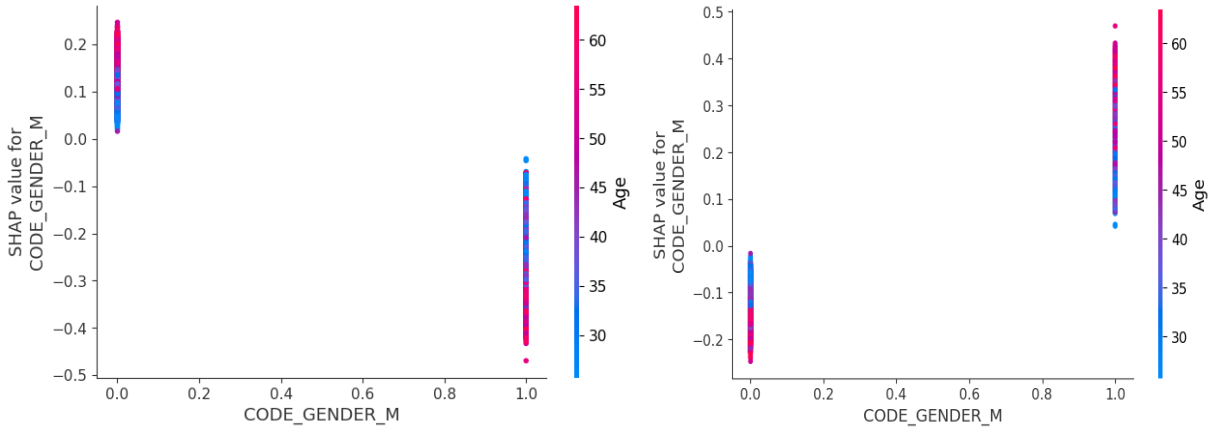
$$\text{Output modèle} = \sum_{i=0}^{i=n} Ei$$



Le signal de sortie de l'interpréteur SHAP, trace la somme des contributions de chaque entité en entrée ayant contribué en sortie du modèle à la valeur de la variable cible. Une distribution en barre horizontale est tracée sur la façon dont il contribue aux résultats ciblés avec plus ou moins de couleurs bleu ou rouge. Autrement dit, il va apporter son poids à l'ensemble des entités du bon ou mauvais côté du seuil de décision d'offre de prêt.

Sur ce graphe, l'entité Ext_Source 2 est la variable qui a le plus de poids, par conséquent, qui a le plus d'impact sur la sortie du modèle.

- Interprétabilité locale : L'interpréteur SHAP analyse et trace la dépendance d'une entité en entrée par rapport à une autre entité. Analysons l'interprétabilité locale de l'entité Genre Masculin sur le risque Crédit en fonction de l'entité Age. La distribution d'Age client dans la base des données est de 20 à 70ans.



Genre Masculin :20 à 35ans : Prêt potentiellement non sécurisé 35 à 50ans : Prêt potentiellement sécurisé 50 à 70ans : Prêt potentiellement non sécurisé

Genre Masculin :20 à 36ans : Prêt potentiellement non sécurisé 35 à 57ans : Prêt potentiellement sécurisé 57 à 70ans : Prêt potentiellement non sécurisé

5 Les limites et les améliorations possibles

5.1 Les limites :

Les limites de notre étude ont été identifiées comme suit :

- La stabilité du modèle de classification au réentrainement.
- Biais lié aux bruits générés par un nombre important de variables .
- Biais lié à des variables insignifiantes

5.2 Les améliorations :

Les améliorations à faire dans notre étude ont été identifiées comme suit :

- Recueillir précisément les objectifs métiers de l'application
- Optimiser le projet avec des modèles de classification en Deep Learning.
- Implémenté un algorithme de surveillance de toute dérive temporelle du modèle .