PROJET 7 IMPLÉMENTEZ UN MODÈLE DE SCORING

PRÉPARÉ PAR
SOULEYMANE CAMARA

MÉMOIRE MÉTHODOLOGIQUE

PRÉSENTATION DE LA DÉMARCHE DE MODÉLISATION

CONTEXTE DE L'ÉTUDE

Ce mémoire constitue l'un des livrables du projet «Implémentez un modèle de scoring » du parcours Data Scientist d'Openclassrooms. Il présente le processus de modélisation et d'analyse du modèle obtenu afin de le rendre plus facilement interprêtable.
Le projet consiste à développer pour la société « Prêt à Dépenser », une société de crédit de consommation, un modèle de scoring de la probabilité de défaut de paiement d'un client.
Les données utilisées pour ce projet sont une base de données de 303 584 clients comportants 50 features (âge, sexe, emploi, logement, revenus, informations relatives au crédit, notation externe, situation familiale, etc.)

MÉTHODOLOGIE D'ENTRAÎNEMENT DU MODÈLE

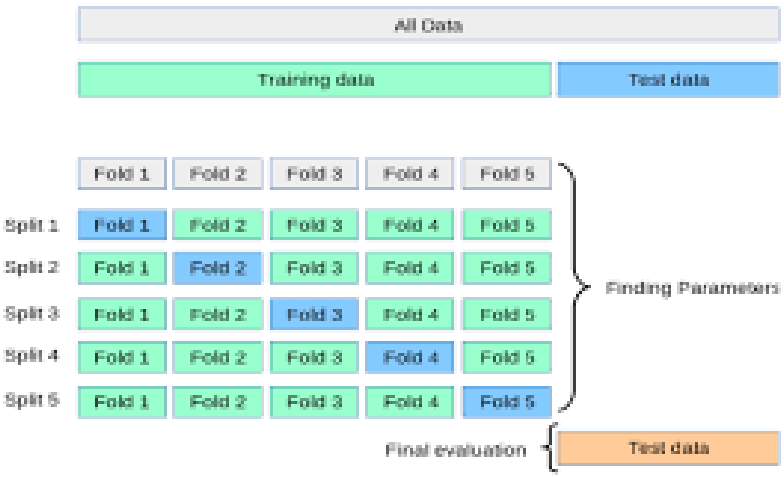
La table initiale contient 122 features, nous avons supprimer les features ayant plus 45 % de valeurs manquantes. Nous passons à **73** variables. Parmi ces 73 variables nous avons 61 features quantitatives et 12 features catégorielles. Sur les 12 catégorielles on a 5 features qui ont deux modalités, que nous avons encodés avec LabelEncoder. Le reste des features catégorielles ont été transformés en utilisant get dummies. Aprés encodage et l'utilisation de get dummies nous obtenons 101 features.

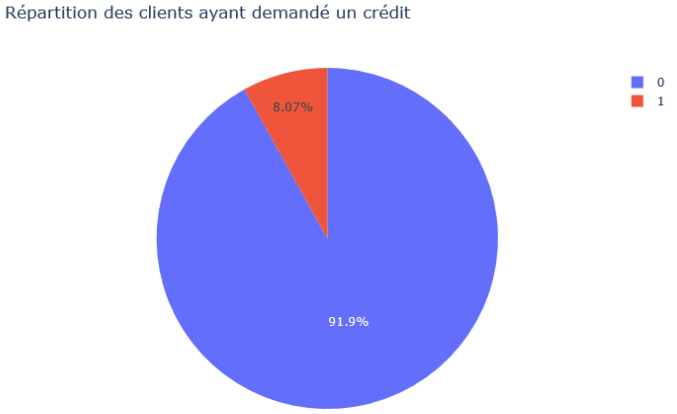
Nous avons crées une fonction polynomiale en utilisant les trois variables suivantes EXT_SOURCE_2, EXT_SOURCE_3, DAYS_BIRTH. Ainsi que la création d'une variable qui donne le pourcentage du montant du crédit par rapport au revenu du client.
Pour la sélection des features pour notre modélisation nous utilisons le module sklearn feature sélection (Threshold, Select kbest, Select From Model) nous obtenons 50 variables pour notre modéle.

Le jeu de données initial a été séparé en plusieurs parties de façon à disposer :

- D'un jeu de training (80% des individus) qui a été séparé en plusieurs folds pour entraîner les différents modèles et optimiser les paramètres grâce à GridSearchCV sans overfitting.
- D'un jeu de test (20% des individus) pour l'évaluation finale du modèle

Nous avons un problème de classification binaire avec une classe sous représentée (8 % de clients qui ne remboursent pas encodé par la valeur 1, contre 91 % de clients qui remboursent encodé par la valeur 0). Ce déséquilibre des classes doit être pris en compte dans l'entraînement des modèles. Deux approches pour rééquilibrer les deux classes sont : Stratify dans le train_test_split et Smote.
SMOTE permet de créer des données synthétiques à partir des données existantes. Les méthodes d'Oversampling fonctionnent en augmentant le nombre d'observations de la (des) classe(s) minoritaire(s) afin d'arriver à un ratio classe minoritaire/ classe majoritaire égal à 1 (223170/223170) .





LES MODÈLES UTILISÉS

Pour résoudre notre problématique de scoring différents modèles ont été testés avec la recherche d'hyperparamètres grâce à la cross validation (5 folds) :

- **La régression logistique en optimisant l'inverse de la force de régularisation ainsi que la norme utilisée pour la pénalisation**
- **Random Forest Classifier en optimisant les paramétres suivants le nombre te la profondeur de l'arbre**
- **le GradientBoosting Classifier en optimisant le nombre de tour de boost et la profondeur de l'arbre**

Le choix du meilleur modèle a été effectué en retenant le modèle avec le meilleur score sur le jeu de test. le score utilisé pour le choix du meilleur modèle n'est pas une des métriques habituelles (précision, recall, fscore) mais une métrique customisée définie pour mieux répondre à la problématique métier.

La fonction coût utilisée pour la recherche des paramètres est dépendante du type de modèle

FONCTIONS COÛT UTILISÉES POUR LA RECHERCHE DE PARAMÈTRES

Pour la régression logistique

La fonction coût d'une régression linéaire est la somme des carrés de la différence entre les valeurs observées et prédites de la variable dépendante.

Mais en cas de régression logistique, la même fonction de coût ne fonctionnera pas car les valeurs réelles de la régression logistique binaire sont 0 et 1. On va utiliser une fonction sigmoïde qui va transformer la fonction linéaire a une probabilité.

L'optimisation de cette fonction est très difficile car elle est non linéaire en termes de probabilité. Nous prenons donc la fonction log de vraisemblance pour optimiser les paramètres du modèle. La fonction de perte de log est la négation de la fonction de log-vraisemblance moyenne.

Random Forest Classifier

On cherche à minimiser l'impureté. Elle correspond a une mesure probabilistique de la performance de l'arbre de décision.Chaque arbre est entraîné sur un sous-ensemble du dataset et donne un résultat. Les résultats de tous les arbres de décision sont alors combinés pour donner une réponse finale. Chaque arbre "vote" (oui ou non) et la réponse finale est celle qui a eu la majorité de vote. Nous avons utilisés l'index d'impureté de Gini. Les features peuvent être évalués pour voir leur impact dans la construction de l'arbre (mesure de Gini).

Gradient Boosting Classifier

On entraine plusieurs classifieurs faibles et on combine les différents classifieurs afin d'obtenir notre classifieur final le plus performant. Pour obtenir ce classifieur final, il nous suffit d'effectuer la somme pondérée des différents classifieurs. Chaque classifieur Gm possède son taux d'erreur sur le jeu de données pondéré. Pour accorder une plus grande importance à ceux qui ont le taux d'erreur global err le plus faible, on pondère notre somme par le poids. Ce poids αm est calculé par minimisation séquentielle additive d'une fonction de perte exponentielle.

l'optimisation des paramètres est faite en utilisant une recherche sur grille (grid search) et que le choix des meilleurs paramètres pour un type de modèle donnée est fait en fonction de la fonction score.

MÉTRIQUE D'ÉVALUATION

La fonction score est utilisé pour l'évaluation de la performance d'un modèle donné.
Pour comparer des classifications on utilise la précision qu'on appelle l'accuracy (la plus part sont prédits positifs et le sont effectivement), le recall ou le rappel qui nous donne le taux de vrais positifs ainsi que la spécifité qui représente le taux de vrais négatifs. Toutes ces mesures sont calculés à partir de la matrice de confusion.
On rappelle quelques notions qu'on retrouve sur la matrice de confusion :
On rappelle aussi que sur scikit learn la classe minoritaire est positif (+) et la classe majoritaire est négative (-).
TP = vrai positif – la classe minoritaire (**ne rembourse pas**) est correctement prédite comme positive
FP = faux positif – la classe majoritaire (**rembourse**) est mal prédite
FN = faux négatif – classe minoritaire (**ne rembourse pas**) mal prédite
TN = vrai négatif – classe majoritaire (**rembourse**) correctement prédite
Une métrique d'évaluation doit être adapter par rapport à la problématique des banques et leurs crédits. On décide d'adopter la stratégie suivante :
 On cherche à minimiser les faux négatifs (erreur de type 2) c'est à dire la classe minoritaire est mal prédite le client ne peut pas rembourser on le prédit qu'il rembourse.
On peut aussi penser à cette stratégie d'éviter un trop grand nombre de faux positifs le client peut rembourser et on le prédit ne peut pas rembourser.

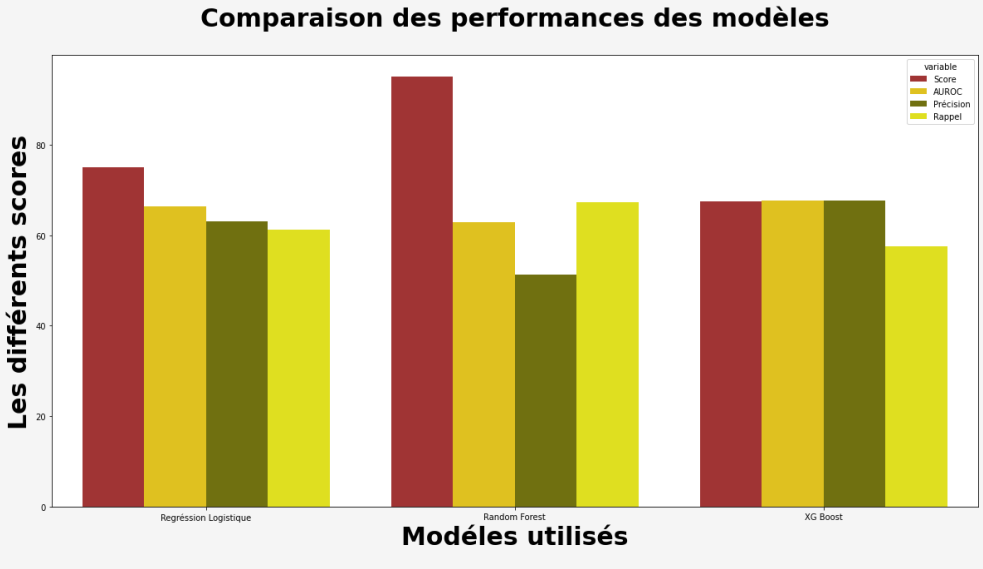
Dans la suite de notre étude on préfère limiter **le risque de perte financier** plus tôt qu'un risque de perte client.
Les deux valeurs qui nous intéressent c'est le FN (la classe minoritaire est mal prédite ne rembourse pas) , le FP (la classe majoritaire est mal prédite il rembourse on le prédit qu'il ne peut pas rembourser) on va faire une combinaison linéaire qui donnera plus de poids au FN car on veut réduire le risque de perte financier.

Ci joint la combinaison linéaire : **$\beta \times FN + (1-\beta) * FP$ avec $\beta \in [0,1]$**

Le choix du meilleur modèle se fait en trois étapes, détaillés ci dessous :

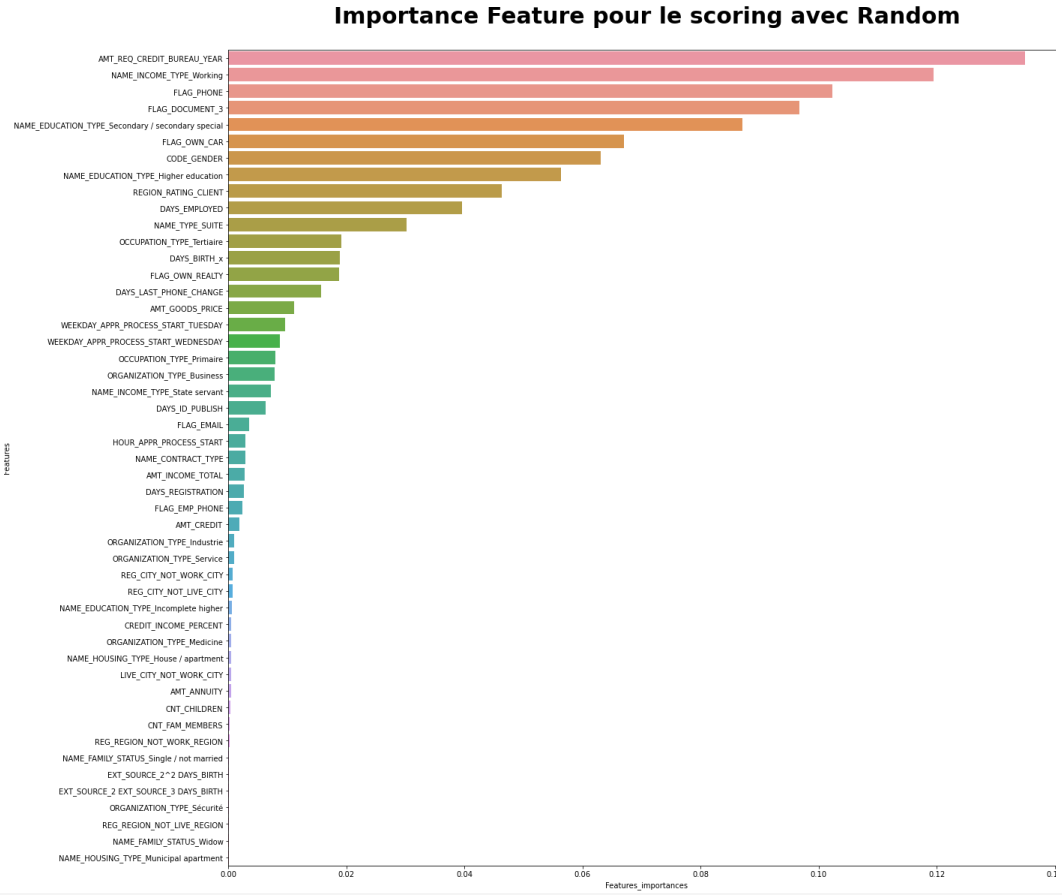
Première étape : La fonction coût est propre au modèle et permet de trouver les meilleurs paramètres pour un paramétrage fixé (lors du balayage du GridSearchCV). Pour choisir le meilleur paramétrage, Grid Search CV va utiliser la fonction score ou une autre métrique de performance. Dans notre cas on va utiliser la fonction score ci dessus.
Deuxième étape : En utilisant la fonction score créée ci dessus nous allons choisir le modèle en prenant pour chaque modèle le meilleur paramétrage issu de l'étape 1
Troisième étape : Le bon modèle sera celui qui aura le meilleur score.

Aprés ces trois étapes nous avons choisis le modéle Random Forest avec les hyperparamétres suivants : le nombre d'arbre (100) et la profondeur de l'arbre (6).



INTERPRÉTABILITÉ DU MODÈLE

La mesure globale de l'importance des variables dépend du type de modèle choisi. Notre modèle Random Forest nous donne l'importance des variables suivantes :



Le modèle étant destiné à des équipes opérationnelles devant être en mesure d'expliquer les décisions de l'algorithme à des clients, le modèle est accompagné d'un module pour expliquer aux clients les raisons d'acceptation ou de rejet de leurs demandes de crédits .

Nous avons des méthodes d'interpretabilité qui permette d'expliquer chaque résultat. L'interprétabilité se définit comme la capacité pour un humain à comprendre les raisons d'une décision d'un modèle. Nous en avons deux SHAPE et LIME avec deux philosophies de fonctionnement différentes.

L'algorithme **LIME** (en anglais, Local Interpretable Model-agnostic Explanations) est un modèle local qui cherche à expliquer la prédiction d'un individu par analyse de son voisinage. Les méthodes locales donnent une interprétation pour un seul ou un petit nombre d'observations.

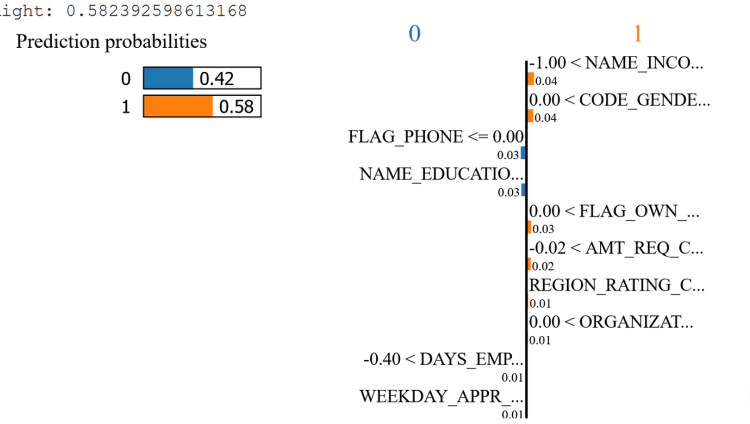
Les différentes étapes de la méthode LIME :

- 1ère étape : l'algorithme LIME génère des nouvelles données, dans un voisinage proche de l'individu à expliquer.
- 2ème étape : LIME entraîne un modèle transparent sur les prédictions du modèle « boîte noire » complexe qu'on cherche à interpréter.

Il apprend ainsi à l'aide d'un modèle simple et donc interprétable (par exemple, une régression linéaire ou un arbre de décision).
Le modèle transparent joue donc le rôle de modèle de substitut pour interpréter les résultats du modèle complexe d'origine.
Le principal inconvénient de la méthode LIME est lié à son fonctionnement local. Et, LIME ne permet pas de généraliser l'interprétabilité issue du modèle local à un niveau plus global.
Au contraire, les méthodes d'interprétation globales permettent d'expliquer toutes les observations en même temps, globalement.

Nous allons utiliser la méthode LIME, pour l'interprétabilité de nos résultats. Ci joint un exemple pour un client avec un identifiant égal à 100038

```
ID client: 100038
Classe réelle : Rejet de la demande de crédit
Intercept 0.8180039790082146
Prediction_local [0.89217185]
Right: 0.582392598613168
```



Feature	Value
NAME_INCOME_TYPE_Working	0.00
CODE_GENDER	1.00
FLAG_PHONE	0.00
NAME_EDUCATION_TYPE_Higher education	0.00
FLAG_OWN_CAR	1.00
AMT_REQ_CREDIT_BUREAU_YEAR	0.00
REGION_RATING_CLIENT	0.00
ORGANIZATION_TYPE_Business	1.00
DAYS_EMPLOYED	-0.30
WEEKDAY_APPR_PROCESS_START_TUESDAY	0.00

		0	1	contribution
0	-1.00 < NAME_INCOME_TYPE_Working <= 0.00	0.040708		normal
1	0.00 < CODE_GENDER <= 1.00	0.036925		normal
2	FLAG_PHONE <= 0.00	-0.030495		normal
3	NAME_EDUCATION_TYPE_Higher education <= 0.00	-0.027280		normal
4	0.00 < FLAG_OWN_CAR <= 1.00	0.025465		normal
5	-0.02 < AMT_REQ_CREDIT_BUREAU_YEAR <= 0.00	0.023088		normal
6	REGION_RATING_CLIENT <= 0.00	0.009915		normal
7	0.00 < ORGANIZATION_TYPE_Business <= 1.00	0.007738		normal
8	-0.40 < DAYS_EMPLOYED <= 0.00	-0.006320		normal
9	WEEKDAY_APPR_PROCESS_START_TUESDAY <= 0.00	-0.005576		normal

LIMITES ET AMÉLIORATIONS POSSIBLES

La modélisation effectuée dans le cadre de ce projet a été effectuée sur la base d'une hypothèse forte :
la définition d'une mérique d'évaluation : création d'une combinaison linéaire entre le FN (faux négatifs) et FP (faux positifs) en mettant plus de poids sur le FN est de limiter les clients qui ne peuvent pas rembourser et qu'on les prédit qu'il peut rembourser. Ce critère minimise la perte financière.
Ce choix de métrique n'est pas confirmé par les gens du métier. L'axe principal d'amélioration serait de définir plus finement la métrique d'évaluation en collaboration avec les équipes métier soit un conseiller bancaire ou un Directeur de Banque.
On peut aussi améliorer le modéle en faisant du feature enginering en collaboration avec les personnes du secteur bancaire. De même sur le choix des features pour la modélisation, on pouvait faire d'autres choix de features en discutant avec les gens du métier.
On peut aussi améliorer les modéles en utilisant les autres tableaux de données à notre disposition pour le calcul d'autres features.

Il est important de trouver un juste compromis en machine learning entre un modèle performant versus un modèle interprétable. En général plus un modèle est complexe plus il est performant et moins il est interprétable.

Avec LIME ou SHAP, on peut rendre interprétable un modèle complexe à l'origine et ainsi favoriser son adoption auprès des métiers. Si comprendre un modèle est un impératif dans toute démarche scientifique, les contraintes juridiques imposent désormais de ne pas prendre une décision basée uniquement sur le résultat d'un algorithme automatique.