Projet 7 : Implémentez un modèle de scoring

Note méthodologique

1. Contexte :

Ce livrable présente le processus de modélisation et d'interprétabilité du modèle mis en place dans le cadre du projet « Implémentez un modèle de scoring ».

Prêt à dépenser est une société financière d'offre de crédit à la consommation pour la clientèle ayant peu ou pas d'historique de prêt.

Notre mission est de développer un modèle de scoring de la probabilité de défaut de paiement du client pour étayer la décision d'accorder ou non un prêt à un client potentiel en s'appuyant sur des sources de données variées.

Le développement d'un dashboard interactif permettra aux chargés de clientèles d'expliquer avec transparence la décision d'octroi ou non du prêt et de mettre à disposition des clients l'exploration de leurs informations personnelles.

2. Jeu de données :

Les données utilisées pour ce projet est une base de données de 307 000 clients comportant 218 features (âge, sexe, revenu, logement, montant prêt …etc.) recueillis auprès Home Credit Group et auprès d'autres institutions financières.

3. La méthodologie d'entraînement du modèle :

3.1 Découpage de jeu de données :

Le modèle entraîné dans le cadre de ce projet a été entraîné sur la base du jeu de données après analyse exploratoire et création de nouvelles features. Le notebook utilisé est consultable sur le site Kaggle (lien dans les notebooks).

Le jeu de données initial a été séparé en deux :

- Un jeu de training (80%) servant à entraîner le modèle.
- Un jeu de test (20 %) pour l'évaluation finale du modèle.

Le problème est un problème de classification binaire avec une classe sous représentée (9 % de clients en défaut contre 91 % de clients sans défaut). Ce déséquilibre des classes doit être pris en compte dans l'entraînement des modèles. Lors de la séparation, les 2 jeux de données devront conserver la répartition de départ des classes majoritaires (les clients non défaillants) et minoritaires (les clients défaillants).

Une classe déséquilibrée peut avoir un impact négatif sur le modèle qui aura tendance à prédire la classe majoritaire (donc client non défaillant).

Une modification de l'ensemble de données est possible avant d'entraîner le modèle prédictif

afin d'équilibrer les données : le rééchantillonnage (re-sampling).

Pour notre modélisation, deux approches pour rééquilibrer les deux classes ont été testées : Class Weight et SMOTE.

Le rééquilibrage via SMOTE ne donnant pas les résultats attendus, le rééquilibrage via l'hyperparamètre Class Weight du modèle LightGBM a été choisi pour le meilleur modèle.

3.2 Choix de modèle :

Nous avons utilisé Pycaret pour lancer les différents modèles de classification installés dans nos environnements pour classer les modèles qui semblent être les plus performants pour notre jeu de données et en prenant en compte les différents scores, sélectionner le modèle le plus prometteur.

Les résultats montrent que les modèles ensemblistes (Catboost, Xgboost et LightGBM) semblent être plus performants sur notre jeu de données. Le modèle LightGBM est très rapide et obtient des résultats satisfaisants qui pourront être améliorés par optimisation des hyperparamètres.

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

4.1 Choix des métriques :

Le choix des métriques permet d'évaluer la performance du modèle prédictif et de garantir la qualité du modèle de classification. Les métriques permettent de comparer les classes réelles aux classes prédites par le modèle.

Pour notre problématique :

- Les défaillants forment la classe positive,
- Les non-défaillants forment la classe négative.

Pour minimiser les pertes d'argent, nous devons :

1- Nous efforcer de ne pas prédire un client non-défaillant s'il est défaillant, c'est à dire, minimiser le nombre de faux négatifs (prédit non-défaillant mais client défaillant).

2- Nous efforcer de ne pas prédire de défaillant si le client n'est pas défaillant donc minimiser les faux positifs (classe 1 défaillant alors que non-défaillant dans la réalité).

Différentes métriques existent pour les modélisations de classification binaire :

Métriques :

- **Recall :** la métrique pour déterminer le taux de vrais positif est le Rappel(Sensibilité)/Recall, elle mesure parmi toutes les observations positives combien ont été classées comme positives. Pour ne pas avoir de pertes, il faut détecter tous les défaillants (classe positive) donc maximiser la métrique recall.

- **Precision :** elle mesure le nombre d'observations prédites comme positives (client défaillant) qui le sont en réalité. Si le client est prédit défaillant alors qu'il ne le sera pas le prêt ne sera pas accordé et les intérêts ne seront pas empochés. Il faut donc maximiser la 'Precision'.

- **F-mesure ou F1 :** compromis entre le rappel et précision.

Si on augmente le recall, la précision diminue, il s'agit de faire un compromis entre ces 2 métriques qui dépendent l'une de l'autre.

Dans notre cas, il faut trouver le plus grand nombre d'observations réellement positives (client prédit défaillant et bien défaillant) et la perte est moins grande si on prédit un client défaillant mais qu'il n'est pas défaillant (faux positifs), donc on donnera priorité à maximiser le recall au dépend de la précision (on parle bien de Precision pas d'accuracy).

Le réglage du paramètre beta pour le Fbeta score permet de donner plus de poids au recall (beta>1) qu'à la précision (0<beta<1). Le score F1 est la moyenne harmonique entre le recall et la précision.

- **Score ROC AUC** : le score ROC AUC est équivalent au calcul de la corrélation de rang entre les prédictions et la cible. Du point de vue de l'interprétation, il est plus utile car il nous indique que cette métrique montre à quel point votre modèle est bon pour classer les prédictions. Elle vous indique la probabilité qu'une instance positive choisie au hasard soit classée plus haut qu'une instance négative choisie au hasard.

- **Score PR AUC :** calcule de l'aire sous la courbe précision-rappel pour obtenir un nombre qui vous donne des informations sur la performance du modèle.

L'application de cette métrique métier passe par la quantification de l'importance

relative entre Recall et Precision, à savoir Beta (β).

Cela revient à estimer le coût moyen d'un défaut, et le coût d'opportunité d'un client refusé

par erreur. Cette connaissance métier n'est pas évoquée à ce stade du projet, nous allons

donc l'estimer. Cette hypothèse pourra bien entendu être modifiée avec un interlocuteur

métier.

$$\text{Beta} = \frac{coef\ Recall}{coef\ Precision}$$

4.2 Optimisation des hyperparamètres du modèle LightGBM :

La technique choisie pour l'optimisation des hyperparamètres du modèle LightGBM est l'optimisation Hyperopt de Pycart.

L'optimisation a été effectuée sur différentes métriques (Roc Auc, PR Auc, F10, Recall) pour différents jeux de données (rééquilibrés avec smote, hyperparamètre class_weight de Lightgbm ou standardisé ou non…).

Le modèle DummyClassifier avec les paramètres de base sert de baseline.

4 L'interprétabilité globale et locale du modèle :

Le modèle étant destiné à des équipes opérationnelles devant être en mesure d'expliquer les décisions de l'algorithme à des clients réels, le modèle est accompagné d'un module d'explicabilité.

L’approche retenue a été d’utiliser SHAP (SHapley Additive exPlanations) qui est une méthode s’appliquant à n’importe quel modèle de machine learning et permettant de comprendre comment évolue la prédiction d’un modèle et perturbant les variables en entrée du modèle.

La méthode SHAP consiste à calculer la valeur de Shapley pour toutes les variables de tous les individus c’est-à-dire la moyenne de l’impact d’une variable (sur la sortie, donc la prédiction) pour toutes les combinaisons de variables possibles. La somme des effets de chaque variable expliquera la prédiction.

Nous avons trouvé que la variable externe EXT_SOURCE_SUM est la caractéristique la plus importante en moyenne.

5 Les limites et les améliorations possibles :

La modélisation effectuée dans le cadre du projet a été effectuée sur la base d’une hypothèse : la définition d’une métrique d’évaluation. Le coefficient Beta pourrait être affiné pour éventuellement améliorer la métrique d’évaluation. Ainsi, une collaboration avec Les experts métiers pourraient nous aider à créer une métrique bancaire plus efficace et adaptée.

Aussi, l’optimisation des hyperparamètres pourra se faire par de larges possibilités, le choix des d’autres méthodes d’optimisation, à savoir optimisation bayésienne, Optuna peut également permettre d’augmenter les performances actuelles.

Par ailleurs,, par faute du temps, la partie de traitement préalable du jeu de données a été abordée de façon superficielle en réutilisant un notebook issu de Kaggle. Il y a très probablement l’opportunité d’améliorer la partie feature engineering et feature selection en utilisant d’autres features des données fournies, ainsi qu’en créant de nouvelles features en collaboration avec les équipes métier.