Projet 7

Implémentez un modèle de scoring

Note méthodologique

Gauthier RAULT

I - La méthodologie d'entraînement du modèle

1 - Preprocessing

Notre mission consiste, en se basant sur les données des clients ayant déjà clôturé leurs prêts (rembourser ou non), à utiliser des algorithmes de machine learning permettant "Prêt à dépenser" (établissements de crédit) de prédire si un prospect sera défaillant.

Afin de faciliter et accélérer la tâche de préprocessing, nous nous sommes inspirés de deux kernel Kaggle avec les liens ci-dessous :

https://www.kaggle.com/willkoehrsen/start-here-a-gentle-introduction

https://www.kaggle.com/danilz/merge-all-data-base-glm-vs-xgb-explained-0-763

Nous avions pour cela à notre disposition 9 datasets (tables) reliés par la clé SK_ID_CURR.

Afin de simplifier notre démarche, nous avons axé notre étude sur deux datasets “application_train.csv” et “application_test.csv” contenant les features dont nous avons besoin.

L’objectif du premier preprocessing fut d’effectuer des jointures afin d’obtenir un unique dataset contenant une ligne pour chacun des clients actuels avec le maximum d'informations sur ce client afin de pouvoir réaliser ensuite des prédictions sur sa capacité de remboursement.

Nous avons également mergé le dataset “bureau” permettant de connaître les prêts clôturés des clients de la base existante.

Dans un second temps, nous avons traité les outliers aberrants, par exemple, la présence des valeurs négatives dans la variable 'DAYS_BIRTH'.

La troisième étape fut l’encodage des variables catégorielles. Nous avons utilisé la fonction “pd.get_dummies()” de Pandas.

La quatrième étape est l’imputation des NaN en utilisant la méthode de SimpleImputer par la médiane.

La cinquième étape est la standardisation des données. Nous avons utilisé la technique "Robust_scaler". Elle utilise l’intervalle interquartile ce qui la rend plus fiable vis à vis des outliers.

La sixième étape a été de remplacer les valeurs infinies par NaN.

2 - Feature engineering

L'objectif a été, à partir des variables existantes, de créer de nouvelles variables pertinentes qui permettraient à l'algorithme d'améliorer ses prédictions.

Nous avons créé 5 nouvelles variables métiers:

• CREDIT_INCOME_PERCENT: Pourcentage du montant du crédit par rapport au revenu d'un client

• ANNUITY_INCOME_PERCENT: Pourcentage de la rente de prêt par rapport au revenu d'un client

• CREDIT_TERM: Durée du paiement en mois

• DAYS_EMPLOYED_PERCENT: Pourcentage des jours employés par rapport à l'âge du client

• INCOME_PER_PERSON: pourcentage des revenus des clients par rapport aux membres de la famille

3 - Equilibrage des classes

Nous avons fait face à une problématique de déséquilibre des classes avec près de 92% des données de la classe négative. C'est une problématique fréquente dans les projets de détection de cas positifs. Ce déséquilibre peut amener l'algorithme à être moins performant dans la détection des cas positifs.

Nous avons utilisé SMOTE qui réalise pour cela un KNN et créé un point à une distance aléatoire d'un point sélectionné et de ses plus proches voisins.

4 – Premier modeling

Afin d'avoir une première idée des performances, nous avons utilisé l'algorithme DummyClassifier permettant d'avoir une baseline.
Puis nous avons balayé différents algorithmes pour obtenir une tendance de ceux à fine tuner.
Nous avons testé 6 algorithmes, en plus de notre baseline et en avons retenu 2.
Le Random Forest et le XGBoost.

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

Hypothèse : Imaginons que nous ayons un capital valant 100.
Faux Négatif = Perte de 50% du capital prêté en moyenne = Perte de 50
Faux Positif = Manque à gagner de 1%/an sur en moyenne 10ans
(capital va de 100 à 0 soit moyenne de 50) = 10% de 50 = Perte de 5
Rapport de coût entre Faux Négatif et Faux Positif:
Les Faux Négatif coûtent 10 fois plus chers que les Faux Positif.
1xFaux Négatif = 10Faux Positif

Nous avons créé une fonction métier fscore avec une affectation de pénalisation des poids pour les Faux Négatif et Faux Positif.

Pour déterminer quelle combinaison d'hyper-paramètres donne les meilleurs résultats sur notre jeu de données, nous avons utilisé la librairie « Hyperopt ».
Elle utilise une approche Bayésienne pour déterminer les meilleurs paramètres d'une fonction. A chaque étape, elle essaye de construire un modèle probabiliste de la fonction et choisit les paramètres les plus prometteurs pour la prochaine étape. Couplé à l'optimisation d'« Hyperopt », un système de validation croisée a été mis en place. Cette technique permet d'éviter le sur-apprentissage (overfitting).

Une fois le modèle choisi et les hyperparamètres optimisés finement d'un point de vue technique via l'AUC, nous avons calculé une fonction de coût métier "fscore". Nous avons réentrainer notre modèle.

Pour rappel :
La precision est la fraction d'exemples vrais positifs parmi les exemples que le modèle a classés comme positifs. En d'autres termes, le nombre de vrais positifs divisé par le nombre de faux positifs plus les vrais positifs.
Le recall, également appelé sensibilité, est la fraction d'exemples classés comme positifs, parmi le nombre total d'exemples positifs. En d'autres termes, le nombre de vrais positifs divisé par le nombre de vrais positifs plus les faux négatifs.

True Positif (TP) est le nombre de vrais positifs classés par le modèle.

False Negative (FN) est le nombre de faux négatifs classés par le modèle.

False Positive (FP) est le nombre de faux positifs classés par le modèle.

Nous avons utilisé 4 métriques par ordre d'importance :

- F-score combine subtilement la précision et le rappel. Il est intéressant, et plus intéressant que l'accuracy car le nombre de vrais négatifs (tn) n'est pas pris en compte. Et dans les situations d'imbalanced class comme c'est notre cas, nous avons une majorité de vrais négatifs qui faussent complètement notre perception de la performance de l'algorithme.

$$F_1 = \frac{2}{\frac{1}{\text{recall}} \times \frac{1}{\text{precision}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$
$$= \frac{\text{tp}}{\text{tp} + \frac{1}{2}(\text{fp} + \text{fn})}$$

- AUC signifie "Area under the ROC Curve"" (aire sous la courbe ROC). C'est-à-dire que l'AUC mesure toute l'aire à deux dimensions sous l'intégralité de la courbe ROC. Ainsi, le modèle le plus efficace a une AUC égale à 1.
- les pourcentages de faux-positifs
- les pourcentages des faux-négatifs

Le Faux-négatif est plus important à éviter puisqu'il signifie qu'un prêt est donné pour un client qui ne peut pas le rembourser, donc la perte touche le capital de la banque. Par contre, le Faux-positif signifie qu'un client qui pourra rembourser son prêt a eu un refus de la part de la banque, donc la perte touche la marge.

Nous avons choisi de ne pas prendre en compte la métrique de ressource calcul car aux vues des sommes engagées pour des prêts, nous estimons que nous aurons un budget en conséquence en face.

III - L’interprétabilité globale et locale du modèle (SHAP values)

Les valeurs SHAP s’appuient sur la théorie des jeux en analysant la contribution marginale de chacune des variables dans la prédiction finale du modèle.

Les valeurs SHAP permettent de décomposer cette différence avec la contribution de chacune des variables. Cela nous permet d’effectuer une analyse locale et globale de l’importance des variables :

- Locale : pour un client donné, nous avons obtenu une explication de la différence avec la prédiction moyenne du modèle comprenant le poids de chacune des variables. Nous savons quelle vairiable a le plus influé la prédiction du prospect

- Globale : pour tout le modèle, nous avons réalisé une moyenne des valeurs absolues des valeurs SHAP pour chacune des variables. Les variables avec la moyenne la plus élevée ont été considérées plus importantes dans les prédictions de manière générale

IV - Les limites et les améliorations possibles

Il serait intéressant d'échanger avec le client et différents départements métiers afin d'améliorer les données choisies et traitées afin d'améliorer l'algorithme et le dashboard.

Compte tenu du temps et de notre connaissance métier du sujet d'étude, nous avons rencontré les limites suivantes :

Nous avions dans une première approche travailler sur un dataset élargi (merger le plus d'information disponible), ce qui a engendré de la complexité, tant sur le nettoyage, les temps de calcul et l'interprétation.
Nous avons fait le choix de travailler sur un dataset plutôt restreint afin de faciliter l'entraînement et la compréhension des résultats.

Échanger avec des métiers afin de comprendre plus en détails la portée des données étudiées, ce qui nous permettra d'améliorer le feature engineering et par extension les résultats de l'algorithme d'apprentissage que nous avons choisi.

Le projet n'avait pas pour focus l'optimisation de cette partie. Suite à des échanges avec le client et une clarification de ses objectifs, cela nous permettrait d'optimiser notre étude pour obtenir des résultats plus en adéquation. Par exemple, nous avons pris l'hypothèse qu'aux vues des sommes engagées concernant un prêt des ressources importantes nous seraient mise à disposition et donc avons écarté le temps de calcul de l'algorithme choisi.

Sur la partie dashboard, nous pourrions augmenter le dynamisme car ce dernier permet de répondre à des questions mais ne permet pas de changer des valeurs à la volée (par exemple si nous avons une augmentation de salaire, l'arrivée d'un enfant, le changement de situation familiale).

Approfondir l'étude du dataset (cleaning, nombre de features sélectionnées, feature engineering), l'étude des algorithmes (nombre, métriques, fine tuning).