OpenClassrooms

Note Méthodologique

Parcours Data Scientist - Projet 7
Implémentez un modèle de scoring

Yann Pham-Van

16/10/2023

Sommaire

Repository GitHub :

https://github.com/Bruce2Cluny191/Projet7-Implementez_un_modele_de_scoring

1. Contexte et mission

Le projet est porté par l'entreprise *Prêt à dépenser*.

L'entreprise, avec son offre de crédits, souhaite développer un algorithme de **classification** indiquant la **probabilité** de défaut de remboursement.

Un **dashboard** interactif permet aux chargés de clientèle l'interprétation des prédictions dans un souci de dialogue transparent avec les clients.

Le modèle de scoring, sous forme d'une **API**, et le **dashboard** interactif doivent être mis en production séparément.

Les données fournies sont accessibles via ce Dataset.

Les livrables proposés à l'issue de ce projet sont hébergés sur la plateforme Heroku :

- API
- Dashboard

2. Modélisation

2.1. Kernel de feature engineering

Un kernel est utilisé afin de gagner du temps sur le préprocessing et le feature engineering du dataset.

Il est disponible sur la plateforme Kaggle :
https://www.kaggle.com/code/jsaguiar/lightgbm-with-simple-features/script

Globalement, il vise à :

- agglomérer les différents dataframes inclus dans le dataset
- encoder les variables qualitatives avec la fonction *get_dummies*
- supprimer 4 observations avec un genre inconnu
- créer de nouvelles variables aggrégées

Pour la modélisation, seules les observations avec indication d'une target sont utilisées pour la construction du modèle.
Les autres serviront à tester le modèle final, avec appel depuis le dashboard.

Après feature engineering, le dataframe de modélisation comprend 796 variables + 1 target.

2.2. Préparation des données

Quelques opérations supplémentaires sont nécessaires avant d'aborder la modélisation :

- uniformisation des entiers au format *int32* pour réduire l'emprise mémoire
- remplacement de quelques valeurs *inf* par des manquants *NaN*

- suppression des variables comprenant trop de valeurs manquantes
- séparation du dataframe en données d'**apprentissage** ou de **test**

2.3. Modélisation

Un modèle naïf va servir de référence : **DummyClassifier**.

Ensuite, différents modèles sont mis en compétition :

- RandomForestClassifier
- LogisticRegression
- XGBClassifier
- LGBMClassifier

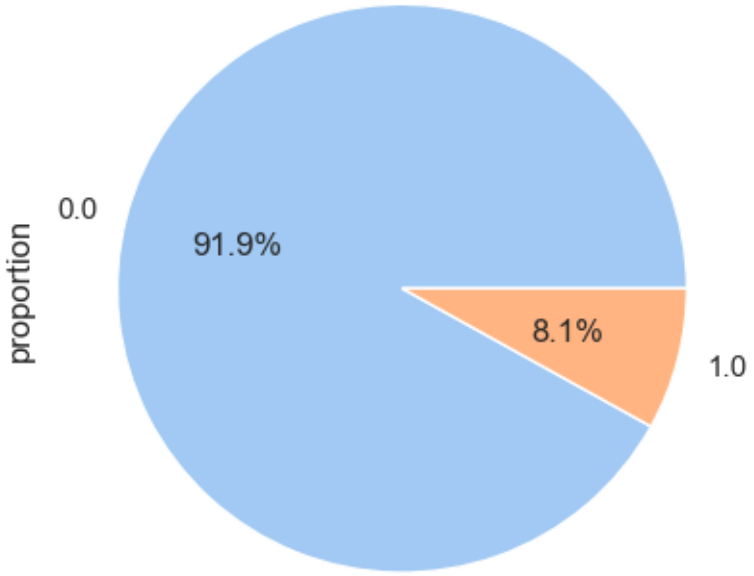
Une recherche d'hyperparamètres - via **GridSearchCV** - définit le meilleur d'entre eux en termes de métriques et de temps de calcul.

Ce dernier est ensuite optimisé à l'aide d'une recherche plus poussée des meilleurs hyperparamètres, via **RandomSearchCV** et l'outil de **tracking** des runs d'expérience de **MLFlow**.

Pour optimiser encore plus le modèle et la balance entre les *faux positifs* (manque à gagner sur les clients refusés mais auraient remboursé) et les *faux négatifs* (prêts accordés à des clients qui vont avoir un défaut de remboursement) qui coûtent 10 fois plus, le **meilleur score métier** est recherché en fonction du **seuil de décision** (*threshold*). Celui-ci est légèrement différent du seuil standard à 0,5.

3. Déséquilibre des classes

Les clients mauvais payeurs sont heureusement beaucoup plus rares que les clients sans défaut de remboursement.



Toutefois, cela implique de tenir compte de ce déséquilibre afin de ne pas biaiser la modélisation.

Différentes techniques sont possibles.

Un essai de la bibliothèque *SMOTE* ne s'est pas révélé fructueux.

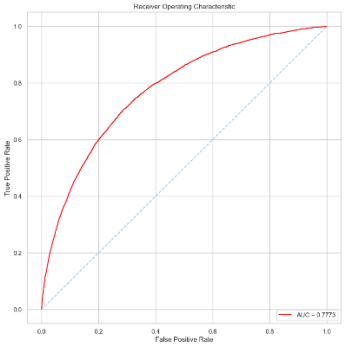
Comme chacun des modèles testés permet la prise en compte d'un déséquilibre via l'argument *class_weight* (ou *scale_pos_weight* pour *XGBClassifier*), cette possibilité simple et performante sera généralisée.

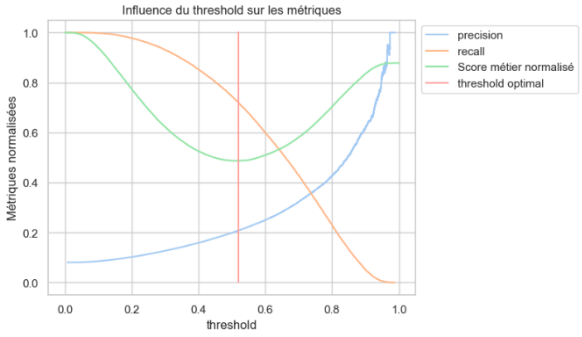
4. Métriques et optimisation

4.1. Métriques

D'une manière générale, la métrique la plus adaptée pour comparer les performances de modèles de classification différents est l'**AUC** (Area Under Curve) :

Toutefois, notre problématique soulève un aspect métier important : la notion de coût.
Ce coût est quantifié par les faux positifs *FP* (manque à gagner) et les faux négatifs *FN* (perte sèche), ces derniers pénalisant 10 fois plus.
Un **score métier** est mis en place = **FP + (10 * FN)**
Evidemment, le modèle recherché doit minimiser cette perte.





4.2. Optimisation

La courbe du score métier ci-dessus est construite par pas de 0,01 pour une vue d'ensemble. Ensuite, une optimisation ultra poussée vise à obtenir le score métier véritablement le plus petit, par une méthode entonnoir, jusqu'à trouver le score qui diminue de seulement **1** (FP) avant d'augmenter de **10** (FN) :

	seuil	score métier
0	0.517823	137485
1	0.517825	137484
2	0.517826	137484
3	0.517827	137484
4	0.517829	137494

5. Résultats

Le tableau suivant résume les métriques obtenues au fil de la recherche du meilleur modèle, puis de son optimisation :

Mode	Modèle	AUC	Score métier	Temps (s)
Paramètres de base	DummyClassifier	0.5000	**73290**	
	RandomForest	0.7235	**73202**	48
	LogisticRegression	0.7722	**46421**	30
	XGBoost	0.7657	**48016**	98
	LightGBM	0.7810	**45502**	27
Optimisation des hyperparamètres par GridSearchCV	RandomForest	0.7324	**51724**	
	LogisticRegression	0.7727	**46175**	
	XGBoost	0.7742	**46829**	
	LightGBM	0.7773	**46083**	
Optimisation du modèle final par RandomSearchCV	LightGBM	0.7846	**44767**	
Recherche du meilleur seuil de décision	LightGBM		**44705**	

Le modèle **LightGBMClassifier** se révèle le plus performant sur les 2 métriques et le plus rapide.

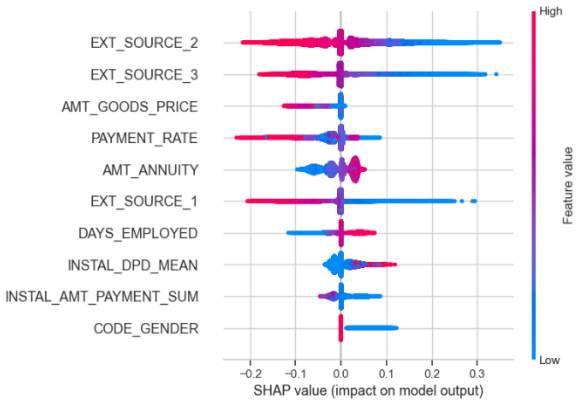
Il est ensuite encore mieux exploité en optimisant ses hyperparamètres et le seuil de décision (threshold).

6. Interprétabilité

Au-delà de la décision d'accorder un prêt, le chargé de clientèle doit être en mesure d'expliquer les raisons menant à cette décision.
Il peut s'appuyer sur la description des variables qui sont le plus impactantes pour l'ensemble des clients (global) et sur les variables du client (local) ayant conduit à la décision finale.

6.1. Interprétabilité globale

Il s'agit là du reflet de l'ensemble des clients existants :

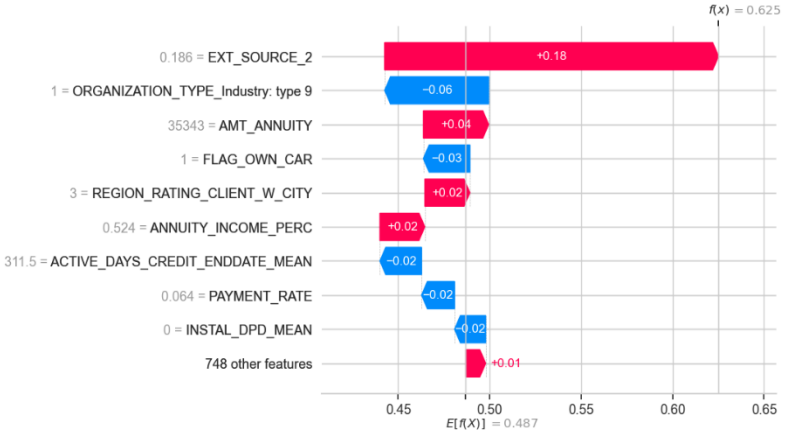


Les *shap_values* ont été déterminées sur l'ensemble du dataframe de modélisation (plus de 307k observations).

Les variables sont classées par influence décroissante, pour cette visualisation *summary*.

6.2. Interprétabilité locale

Il s'agit ici d'un exemple pour un nouveau demandeur :



Les *shap_values* sont fixées depuis un échantillon.

La ligne d'un demandeur précis est appelée pour cette visualisation *waterfall*.

7. Limites et perspectives

Le score métier est basé sur la minimisation des coûts de non conclusion de prêts valables (**FP**) et d'accords de prêts aboutissant à des défauts de paiement (**FN**).

Il serait aussi intéressant de tenir compte des **vrais négatifs** (**TN** = les emprunts accordés qui se déroulent sans accroc) qui viennent compenser - et dépasser, il faut l'espérer - les coûts précédents.

Si la valeur (**X**) d'un tel **bon client** était connue, le score métier pourrait en tenir compte pour une fonction qu'on chercherait à **maximiser** cette fois :

(X * TN) – FP – (10 * FN)

Nota : les vrais positifs (**TP**) n'interviennent pas dans le score car étant détectés, ils ne coûtent rien mais ne rapportent rien non plus !

8. Data Drift

Le modèle final ne l'est pas tant que ça !

Au fil du temps, la population évolue et ses caractéristiques aussi.

Le modèle est donc susceptible d'être moins pertinent et nécessiterait une mise à jour, du fait d'une distribution des variables ayant évolué.

La mesure du **data drift** permet de surveiller le moment où cela devient nécessaire.

La bibliothèque ***evidently*** analyse un tel changement des distributions entre une population de référence, celle ayant servie à l'apprentissage du modèle, et une population réactualisée, ici définie par le **dataframe sans target** mis de côté au §2.1.

Le rapport indique une évolution notable des distributions pour 7,438%, soit 9 des 121 variables initiales (avant feature engineering), ce qui permet de conclure globalement à une **absence de data drift**.

Toutefois, sur les 9 variables concernées, celles-ci méritent une attention particulière à l'avenir :

- montant du bien
- montant du crédit sollicité
- montant du remboursement annuel