NOTE MÉTHODOLOGIQUE

Alexandre Delaguillaumie

LA MÉTHODOLOGIE D'ENTRAÎNEMENT DU MODÈLE

- Au préalable, le feature engineering, le preprocessing et le merge des tables provient du kernel Kaggle, que vous pouvez retrouver ICI. Un traitement des outliers pointant notamment des infinis à été fait et les index contenant plus de 45% de NaN on été supprimé. On note également un désequilibre des classes de l'ordre de 91,9%.

- Un sampling a été fait pour accélérer les temps de calcul ainsi qu'un train_test_split (proportion 80/20). Ce à quoi s'ajoute le traitement du déséquilibre des classes. Trois ensembles de splits ont été réalisés ; un premier avec les imbalanced datas, un deuxième avec une stratégie d'oversampling via SMOTE pour ajouter artificiellement des index de la classe minoritaire et undersampling qui est la contraposée de l'oversampling, via le Random Under Sampling (RUS). Chaque ensemble sera testé sur chaque algorithme.

- Concernant la partie entrainement du modèle, j'ai utilisé un GridSearchCV avec une cross-validation de 5, un random_state et j'ai enregistré les métriques, paramètres et scores dans MLFlow a chaque itération.

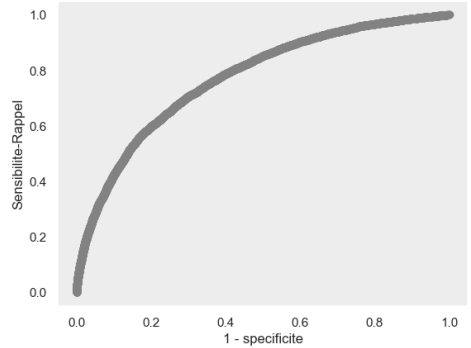
FONCTION COÛT MÉTIER

FN = -10

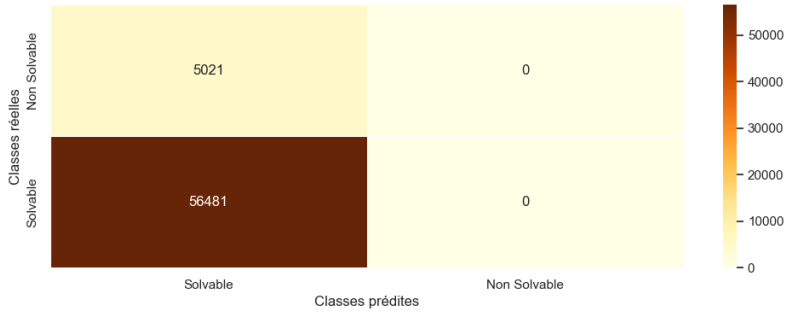
TN = +1

Nous sommes face à une problématique d'octroi de crédit. Dans ce cas de figure, la gravité d'une erreur de prédiction dépend de là ou l'on place la priorité. Si l'entreprise considère qu'il faut en priorité détecter les clients non solvables, on maximisera les vrais positifs (dans le cas d'un dataset déséquilibré il suffit d'un Dummy model à classe majoritaire). Si au contraire elle décide qu'il ne faut pas passer à côté d'un client solvable, il faut maximiser les vrais négatifs. Ici on a considéré que la position dans laquelle l'entreprise ne veut pas se retrouver est celle ou elle passe à côté d'un client solvable, qui peut devenir un énorme manque à gagner sur l'ensemble des clients vu le peu de clients non-solvable. Ainsi il convient de minimiser les faux négatifs et diminuer les risques en maximisant légèrement la détection des personnes réellement solvables. Cette fonction va donc attribuer des poids à chaque catégorie de prédiction dans la matrice de confusion et retourner la fonction d'évaluation déterminée par les gains normalisés. Cette fonction coût métier est bien évidemment à affiner selon les exigences économiques et éthiques de l'entreprise.

AUROC



MATRICE DE CONFUSION

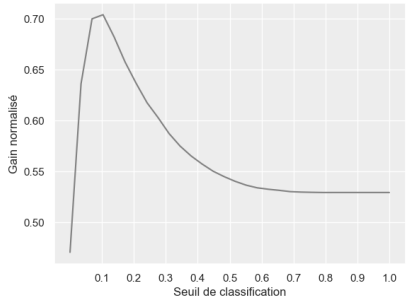


Pour mesurer la pertinence de la fonction coût métier, il convient de déterminer la bonne métrique. Ici, c'est la specificity (taux d'individus négatifs correctement prédit) et, dans une moindre mesure, la precision (capacité du modèle à ne pas faire d'erreur lors d'une prédiction positive) qui prévalent. Ce sont donc les roc_auc et le F1 score qu'il faut observer.

MÉTRIQUES

rocauc,
f1 score,
make_scorer

FONCTION COÛT



ALGORITHME ET OPTIMISATION

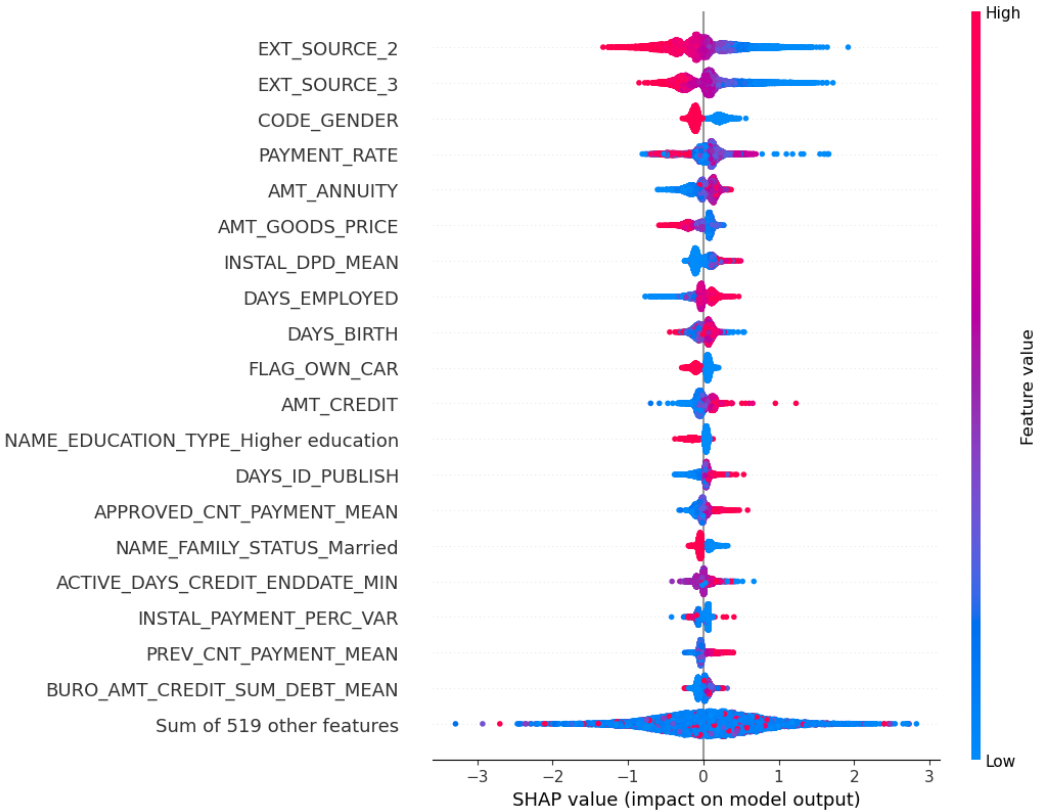
XG Boost

J'ai tout d'abord entrainé un Dummy Model, qui nous informe que le minimum à espérer est un auroc de 0.50. Parmi Random Forest, Regression Logistique, K Neighbors, et XGBoost, le modèle qui enregistre les meilleures performances est XGBoost. Il est optimisé avec les données under-samplées, avec un score auroc de 0.672. La prédiction a été améliorée en définissant manuellement un threshold (ici autour des 0.1). Pour le définir, j'ai testé une trentaine de seuils de classification différents et choisi celui pour lequel la fonction coût métier était optimal. Ainsi la fonction coût arrive aux alentours des 0.70.

TABLEAU DE SYNTHÈSE DES RÉSULTATS

					Metrics				
	Run Name	Created	Duration	Experiment Name	accuracy_score	balance_accuracy	f1	fbeta_score	roc_auc
	xgb_threshold/0.10344827586206896	4 days ago	8.8s	XGBoost Models	0.233	0.567	0.171	0.5	0.567
	xgb_threshold/0.06896551724137931	6 hours ago	6.4s	XGBoost Models	0.742	0.7	0.292	0.522	0.7
	xgb_final/rus/2	10 hours ago	2.0h	XGBoost Models	0.676	0.684	0.261	0.521	0.684
	xgb_final/rus/1	4 days ago	1.2h	XGBoost Models	0.675	0.672	0.251	0.502	0.672
	xgb/smote	4 days ago	49.2min	XGBoost Models	0.917	0.505	0.024	0.014	0.505
	xgb/rus	4 days ago	6.6min	XGBoost Models	0.678	0.678	0.256	0.51	0.678
	xgb/imbalanced	4 days ago	22.4min	XGBoost Models	0.919	0.516	0.066	0.039	0.516
	rf/smote	4 days ago	58.0min	RandomF Models	0.89	0.55	0.176	0.149	0.55
	rf/rus	4 days ago	4.5min	RandomF Models	0.68	0.676	0.256	0.507	0.676
	rf/imbalanced	5 days ago	41.1min	RandomF Models	0.918	0.5	0	0	0.5
	logistic_regression/smote	5 days ago	1.4min	LogReg Models	0.73	0.652	0.253	0.449	0.652
	logistic_regression/rus	5 days ago	17.5s	LogReg Models	0.663	0.658	0.24	0.486	0.658
	logistic_regression/imbalanced	5 days ago	41.6s	LogReg Models	0.918	0.541	0.151	0.097	0.541
	knn/smote	5 days ago	21.0min	KNeighbors Models	0.093	0.505	0.152	0.473	0.505
	knn/rus	5 days ago	38.7s	KNeighbors Models	0.773	0.607	0.227	0.352	0.607
	knn/imbalanced	5 days ago	6.7min	KNeighbors Models	0.918	0.5	0	0	0.5
	dummy/uniform	4 days ago	4.7s	Dummy Models	0.507	0.507	0.144	0.337	0.507
	dummy/uniform	4 days ago	9.6s	Dummy Models	0.507	0.507	0.144	0.337	0.507
	dummy/uniform	4 days ago	7.5s	Dummy Models	0.507	0.507	0.144	0.337	0.507
	dummy/prior/smote	5 days ago	4.7s	Dummy Models	0.918	0.5	0	0	0.5
	dummy/prior/rus	5 days ago	4.7s	Dummy Models	0.918	0.5	0	0	0.5
	dummy/prior/imbalanced	5 days ago	5.5s	Dummy Models	0.918	0.5	0	0	0.5

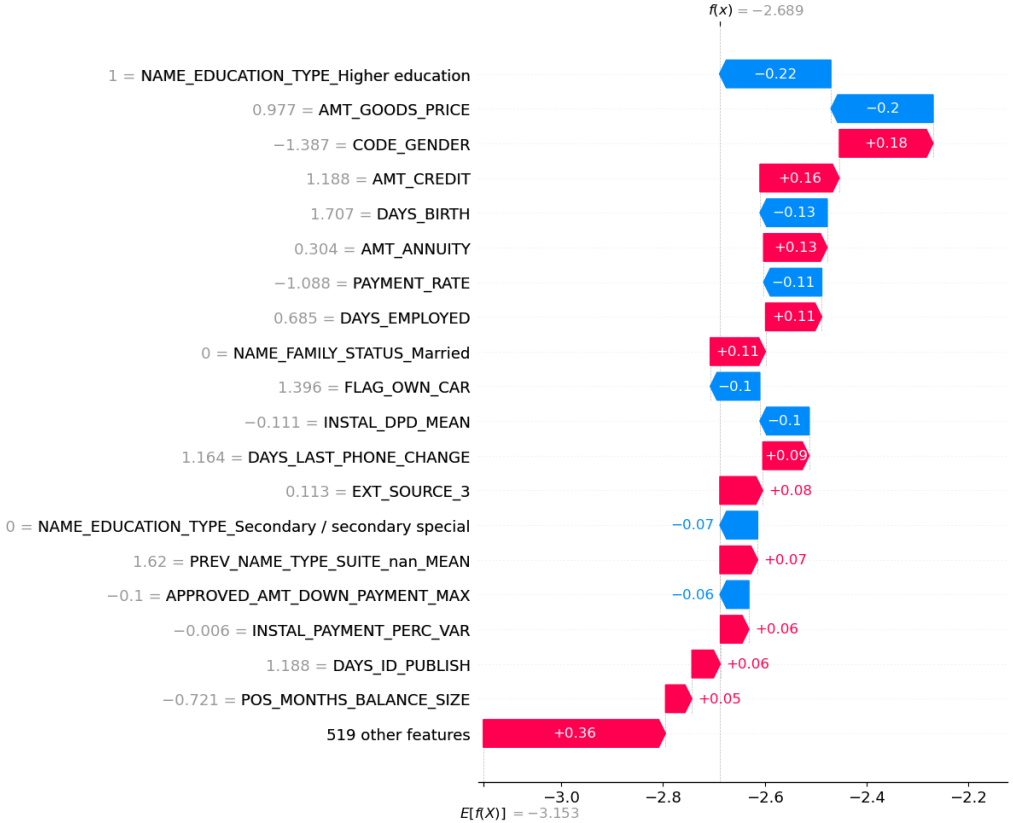
INTERPRÉTABILITÉ GLOBALE DU MODÈLE



Shap est utilisé à des fins d'interprétabilité dans le cadre d'un déploiement du dashboard et permet aux chargés d'études d'expliquer aux client les raisons d'un refus.
L'utilisation spécifique de cette librairie permet de pouvoir visualiser nativement la feature importance et d'avoir une interpretabilité globale et locale, permettant une meilleure contextualisation (et étant adaptable à tout algorithme)

Interpretabilité globale

INTERPRÉTABILITÉ LOCALE DU MODÈLE



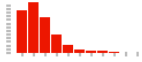
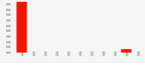
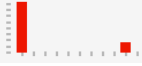
Interpretabilité locale

n°162308

ANALYSE DU DATA DRIFT

Dataset Drift is NOT detected

TESTS	SUCCESS	WARNING	FAIL	ERROR
347	326	0	21	0

Column	Type	Reference Distribution	Current Distribution	Data Drift	Stat Test	Drift Score
AMT_REQ_CREDIT_BUREAU_QRT	num			Detected	Wasserstein distance (normed)	0.359052
AMT_REQ_CREDIT_BUREAU_MON	num			Detected	Wasserstein distance (normed)	0.281765
AMT_GOODS_PRICE	num			Detected	Wasserstein distance (normed)	0.210785
AMT_CREDIT	num			Detected	Wasserstein distance (normed)	0.207334
AMT_ANNUITY	num			Detected	Wasserstein distance (normed)	0.161102
AMT_REQ_CREDIT_BUREAU_WEEK	num			Detected	Wasserstein distance (normed)	0.15426
NAME_CONTRACT_TYPE	cat			Detected	Jensen-Shannon distance	0.14755
DAYS_LAST_PHONE_CHANGE	num			Detected	Wasserstein distance (normed)	0.138977
FLAG_EMAIL	num			Detected	Jensen-Shannon distance	0.122121

DATADRIFT THRESHOLD	0.5
DRIFTED COLUMNS	7.5%

LIMITES ET LES AMÉLIORATIONS POSSIBLES

+

- Il faudrait communiquer un brief plus précis afin d'évaluer précisément la loss function
- Réaliser séparément une black box pour mieux séparer les classes serait intéressant
- Éventuellement réaliser le projet avec Pyspark pourrait améliorer les temps de calcul

-

- La taille du dataset entraine des problème de stockage et de mémoire conséquents.
- L'optimisation des hyperparamètres est très longues et nécéssite une courte liste d'hyperparams
- Les poids de pénalités de la fonction coût sont fixés arbitrairement et empêche l'optimisation
- La demande d'interpretabilité empêche de pouvoir réduire la dimension et améliorer les performances