Note Méthodologique

Projet 7 : Implémentez un modèle de scoring

Mitra Dadgar

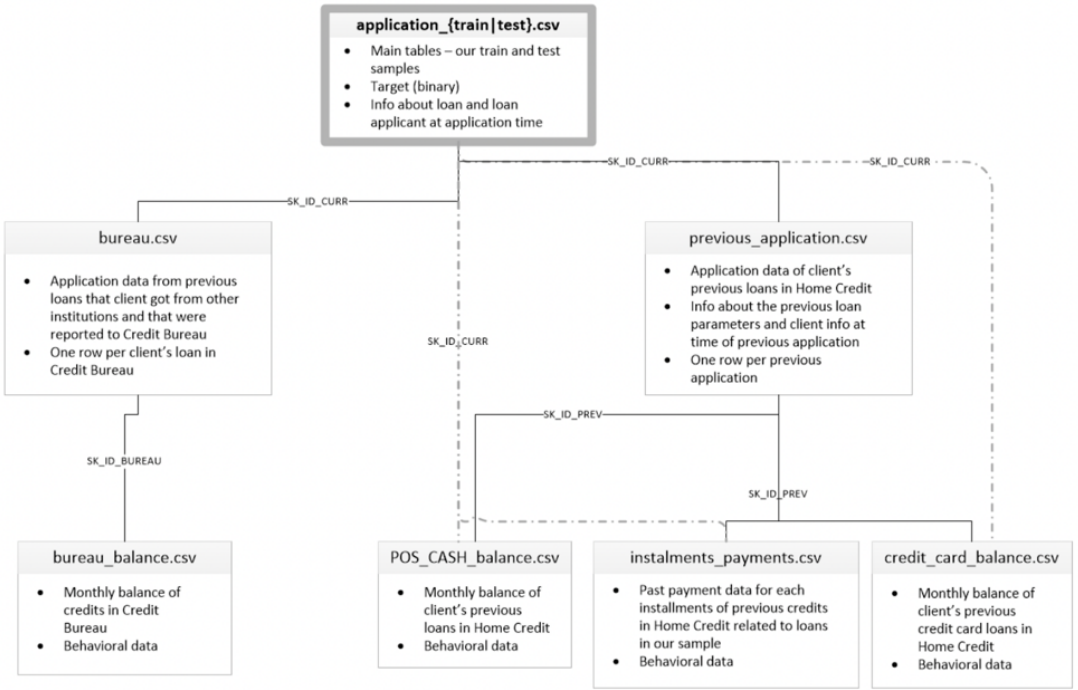
1.Context

La société financière nommée "Prêt à dépenser" propose des crédits à la consommation pour des personnes ayant peu ou pas d'historique de prêt. Cette entreprise souhaite développer un outil utilisant un modèle de scoring permettant d'obtenir la probabilité de défaut de paiement afin de décider si on peut accorder ou non un prêt à un client potentiel en s'appuyant sur des sources de données variées (données comportementales, données provenant d'autres institutions financières, etc.)

Prêt à dépenser décide donc de développer un Dashboard interactif pour que les chargés de relation client puissent à la fois expliquer de façon la plus transparente possible les décisions d'octroi de crédit, mais également permettre à leurs clients de disposer de leurs informations personnelles et de les explorer facilement.

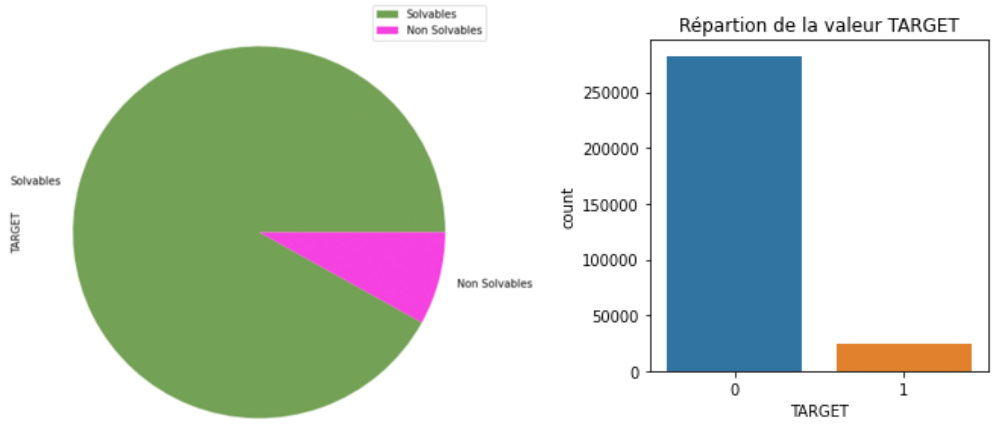
2. Méthodologie d'entraînement du modèle

Sept tables de données sont mises à disposition pour mener à bien notre démarche. Elles sont constituées de données anonymisées d'informations personnelles et bancaires des clients. Les données utilisées pour ce projet sont une base de données de 307 000 clients comportant 121 features (âge, sexe, emploi, logement, revenus, informations relatives au crédit, notation externe, etc.). Base de données principale : application train avec 121 features.




```
app previous Shape: (1670214, 37)
installments payments Shape: (13605401, 8)
sample submission Shape: (48744, 2)
bureau Shape: (1716428, 17)
bureau balance Shape: (27299925, 3)
POS CASH balance Shape: (10001358, 8)
credit card balance Shape: (3840312, 23)
```

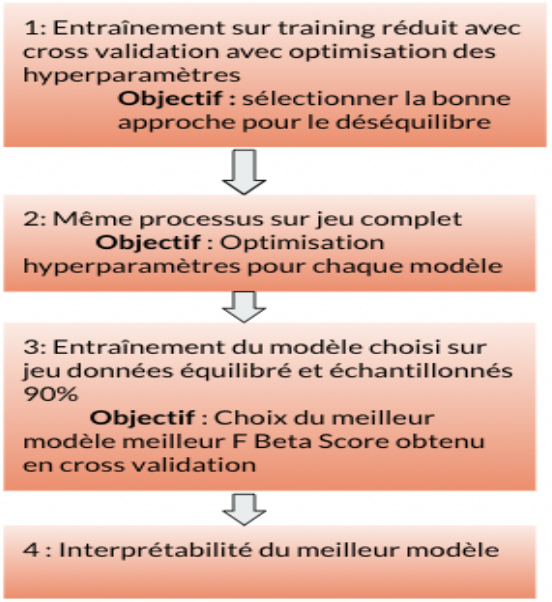
On remarque tout d'abord, un déséquilibre. La valeur à prédire est contenue dans la variable « TARGET ». Les données du jeu d'entraînement sont déséquilibrées : environ 92 % de TARGET = 0" pour 8 % de "TARGET = 1. 1 qui indique que le client n'a pas remboursé son crédit. (Défaillant, 8%) - 0 qui indique que le client l'a remboursé. (Non défaillant, 92%)



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.

Nous transformons des données, Les variables catégorielles doivent être encodées pour être utilisables par les modèles. Nous faisons un encodage des données catégorielles (One Hot Encoder) et un redimensionnement des données (StandardS caler). Ensuite nous avons pris un échantillon d'un premier coup 30% et deuxième fois 90%. 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 (cross validation) sans overfitting. ; D'un jeu de test (20 % des individus) pour l'évaluation finale du modèle.

Choix de sept classifieurs: Dummy Classifier(naïve/baseline), Decision Tree, Random Forest, Logistic Regression, Xgboost, LightGBM, Catboost. Étapes de méthodologie d'entraînement du modèle :



3. Fonction cout, algorithme d'optimisation et métrique d'évaluation

Les modèles ont été entraînés dans la fonction de cross validation et testés suivant différentes combinaisons des hyperparamètres.

model	best_param
DummyClassifier	{'strategy': 'most_frequent'}
DecisionTree	{'criterion': 'entropy', 'max_depth': 5, 'max_features': None}
RandomForest	{'bootstrap': True, 'max_depth': 5, 'n_estimators': 100}
LogisticRegression	{'C': 1.0, 'max_iter': 200, 'penalty': 'l2'}
XGB	{'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 100}
LGBM	{'learning_rate': 0.0001, 'max_depth': 5, 'n_estimators': 100}
CatBoost	{'depth': 5, 'iterations': 20, 'learning_rate': 0.01}

Nous recherchons un scoring adapté au problème du métier. Pour la classification binaire, les métriques pour estimer les erreurs entre y_pred et y_test sont :

Précision : Quelle portion du Target prédit sont de la vraie classe ? Minimiser les faux positifs

Précision = TP / (TP + FP)

ROC AUC (Area Under the Curve) : peut être comparé entre modèles

Recall : Quelle partie de la vraie classe est présente dans la classe prédit ? Minimiser les faux négatifs

Recall = TP / (TP + FN)

F1-score : Accuracy équilibré, Généralisation de F1-score pour mettre plus de poids sur précision, pour mettre plus de poids sur Recall (ex : beta=2)

F1 = 2 (précision recall) / (précision + recall) F 1 = (2 TP) / (2 TP + FP + FN)

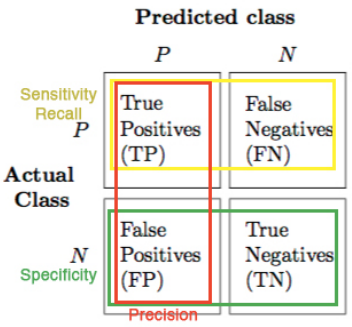
Fbeta-score : Moyenne harmonique pondérée

((1 + β^2) * précision * recall) / (β^2 * précision + recall)

Il y 3 types de coûts à évaluer, la fonction de score est une somme des coûts dont l'objectif est d'être maximiser. Problématique : La société ne doit pas se priver des potentiels clients qui ne présentent pas de risque et en plus les clients à risque font perdre de l'argent à la société donc la banque peut commettre 2

erreurs : Refuser un prêt dû Perdre le client et perdre le client (FP), accorder un prêt indu Perdre la somme prêtée (FN).

Matrice de confusion		Classe prédite	
		Classe 0 : non défaillant	Classe 1 : défaillant
Classe réelle	Classe 0 : non défaillant	Vrais négatifs TN	Faux positifs FP
	Classe 1 : défaillant	Faux négatifs FN	Vrais positifs TP

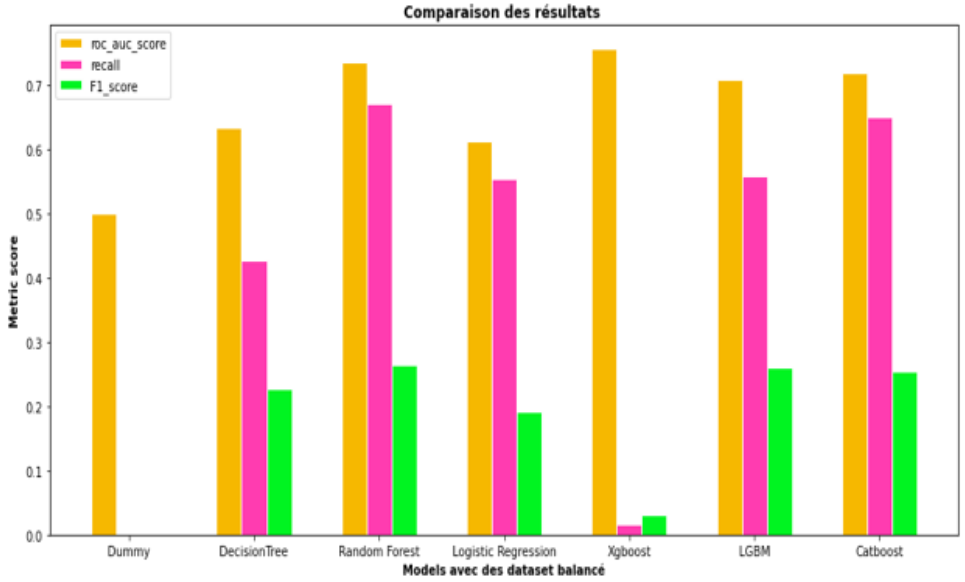


Nous avons deux solutions, limiter le nombre de faux négatifs et limiter dans une moindre mesure le nombre de faux positifs. Explication fonction métier :

TP	Pas de gain, pas perte ni de gain
TN	Gain des intérêts du crédit
FP	Perte des intérêts du crédit
FN	Perte des intérêts du crédit et une partie du crédit

Pour déterminer l'algorithme optimal, La meilleure combinaison des hyperparamètres a été retenue pour chaque algorithme. Nous ajoutons un hyperparamètre « class_weight » que l'on peut régler sur « balanced » afin qu'il tienne compte du déséquilibre des classes. Les modèles ayant le meilleur score en cross validation sur le jeu de training ont été retenus : il s'agit des modèles LightGBM et XGBoost .

	ROC AUC_balanced	Recall_balanced	F1-Score_balanced
DummyClassifier()	0.5	0.0	0.0
DecisionTreeClassifier(class_weight='balanced')	0.6337	0.4276	0.2269
RandomForestClassifier(class_weight='balanced')	0.7366	0.6724	0.2648
LogisticRegression(class_weight='balanced')	0.6131	0.5545	0.1915
XGBClassifier(class_weight='balanced')	0.7565	0.016	0.0311
LGBMClassifier(class_weight='balanced')	0.7085	0.5596	0.2614
<catboost.core.CatBoostClassifier object at 0x7f31a29db510>	0.7191	0.65	0.2542



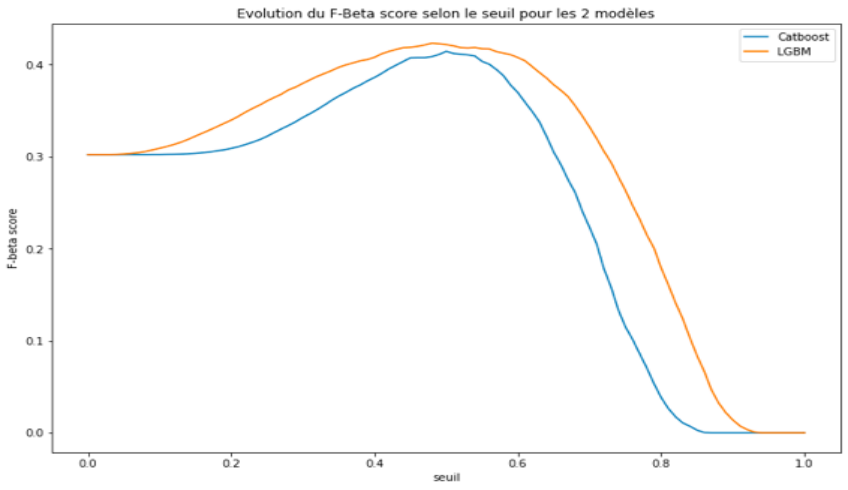
Choix d'une métrique de performance adaptée : utilisation du Fbeta score (beta=2) pour relancer la recherche des hyper paramètres.

Décision : Perte de 10% (perte d'un client) vers Perte maximum de 100%. Nous réapprenons les meilleurs modèles sur le train set complet

model	best_param_df2	roc_auc_df2	f1_score_df2	recall_score_df2	fbeta_score_df2
LGBMClassifier	{'learning_rate': 0.1, 'max_depth': 10, 'n_est...	0.763	0.271	0.675	0.423
CatBoostClassifier	{'depth': 9, 'iterations': 100, 'learning_rate...	0.749	0.259	0.677	0.412

Le seuil de solvabilité optimum, en fonction de notre métrique fbeta2 score

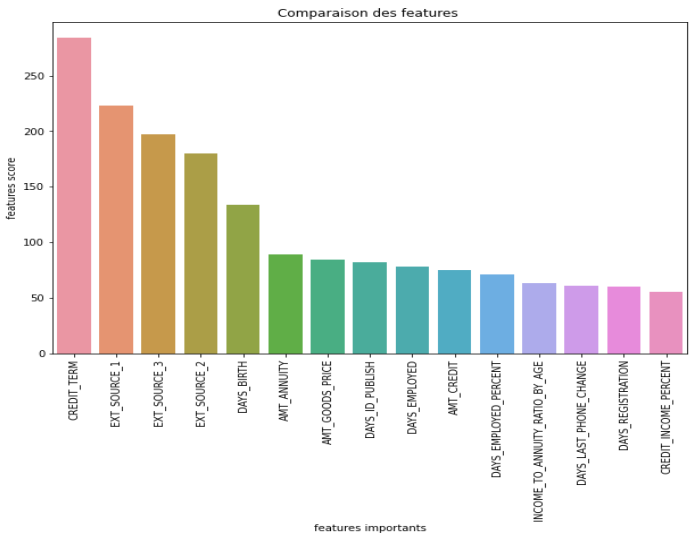
- Score max LGBM: 0.4233
- Seuil pour score max LGBM: 0.48
- Score max Catboost: 0.4144
- Seuil pour score max Cat Boost: 0.5



4. Interprétabilité du modèle

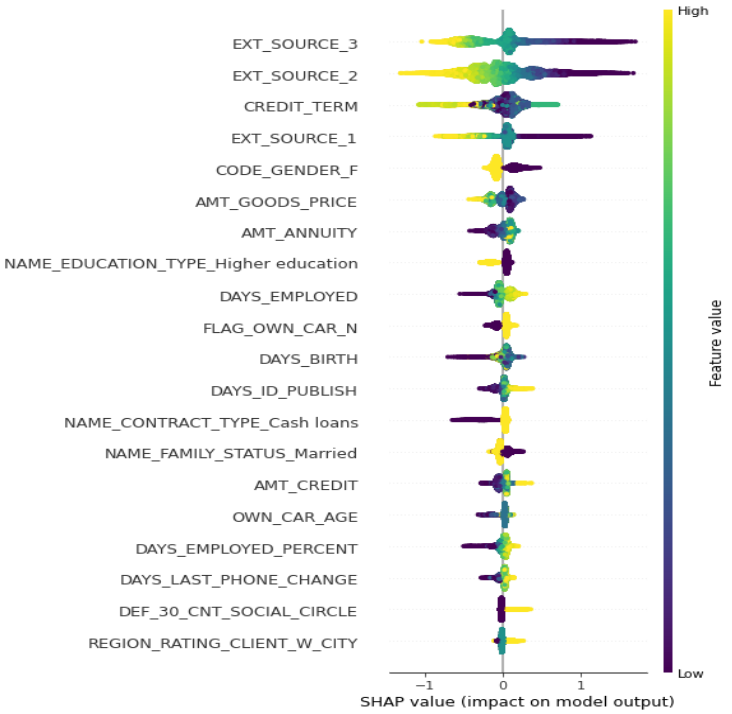
Pour interpréter un modèle, la première perspective envisagée était d'utiliser l'importance des features issues des différents modèles utilisés mais cette approche n'est pas optimale parce que les features importantes en sortie de modèle sont difficiles à interpréter lorsqu'il y a des variables issues de One Hot Encoding.

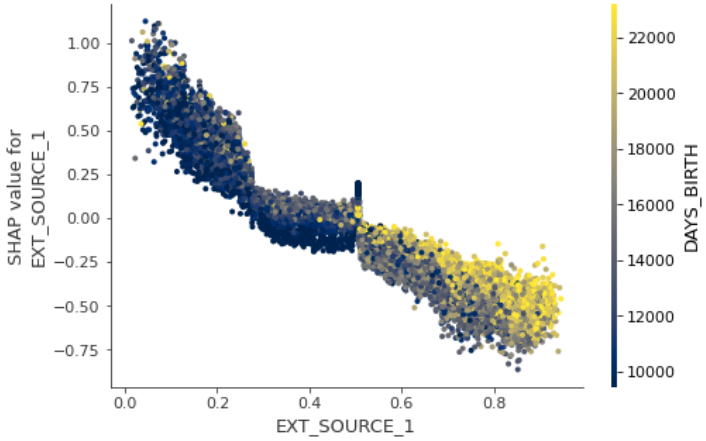
Les variables que nous avions détectées durant l'analyse exploratoire comme fortement corrélées à la cible se trouvent également dans la liste : les variables sources extérieures (EXT SOURCE_2, 3, VAR), les variables âge (DAYS BIRTH), l'ancienneté (DAYS EMPLOYED).

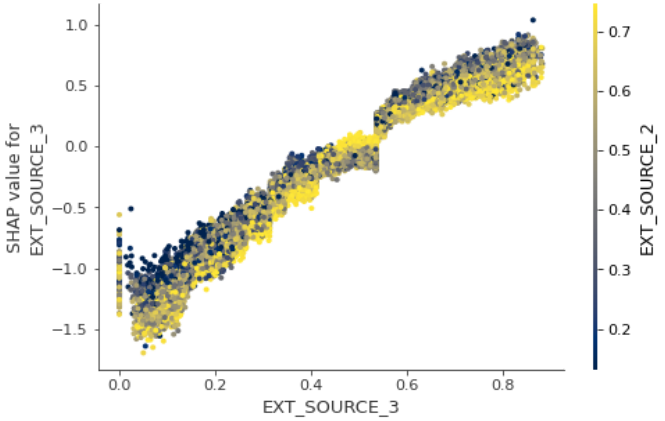


feature	col_score
AMT_CREDIT	84
AMT_ANNUITY	95
AMT_GOODS_PRICE	90
DAYS_BIRTH	140
DAYS_EMPLOYED	67
DAYS_REGISTRATION	56
DAYS_ID_PUBLISH	101
EXT_SOURCE_1	209
EXT_SOURCE_2	177
EXT_SOURCE_3	193
DAYS_LAST_PHONE_CHANGE	69
INCOME_TO_ANNUITY_RATIO_BY_AGE	58
CREDIT_TERM	323
DAYS_EMPLOYED_PERCENT	57

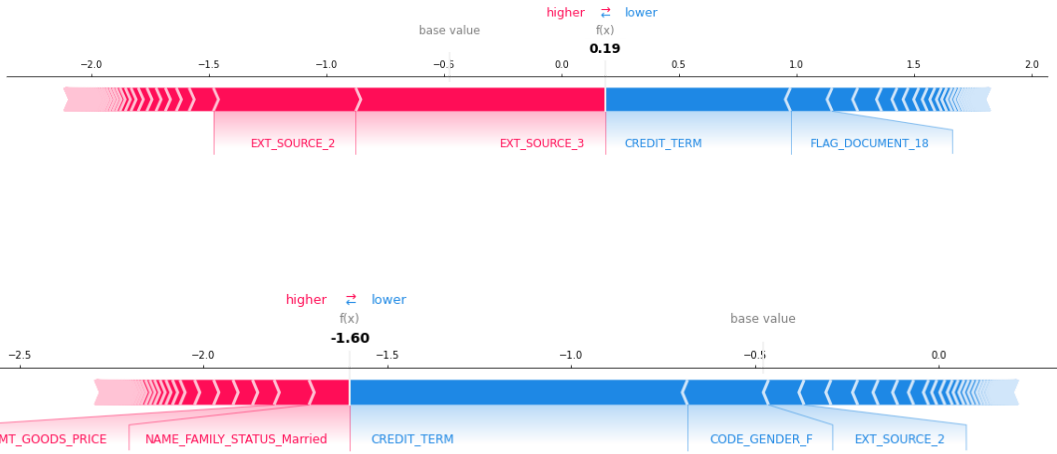
La méthode des SHAP values permet de quantifier l'impact positif ou négatif des différentes features sur les individus. Lorsque l'on regarde globalement l'importance des features avec l'outil SHAP, nous retrouvons les features déterminées précédemment par la fonction intégrée dans notre modèle ainsi que les features représentant le sexe du client et le prix du bien acheté par des prêts de consommation (AMT_GOODS_PRICE), variables discriminantes dans le modèle. Pour EXT SOURCE_2, EXT SOURCE 3, on peut voir que de faibles valeurs augmentent de manière significative la sortie de probabilités du modèle et donc le fait d'être insolvable et le non accord du prêt.







Force plot : pour un client non défaillant (Classe 0) on observe que les variables status de famillial et sexe et prix du bien déterminantes pour indiquer que le client appartient à la classe 0.



5. CONCLUSION _ Les limites et les améliorations possibles

Notre étude portait sur un problème de classification binaire présentant un déséquilibre de classe. Nous avons créé de nouvelles variables en veillant à ce qu'elles restent facilement explicables et nous nous conformons ainsi à la demande de notre client. Ensuite nous avons mis en œuvre des modèles différentes (Dummy Classifier, Decision Tree, Random Forest, Logistic Regression, Xgboost, LightGBM, Catboost) et des stratégies GridSearchCV pour trouver hyperparamètres et optimiser le meilleur modèle et obtenir une performance maximale. Le rééchantillonnage des données permet de corriger le déséquilibre des classes. Utilisation d'une méthode d'échantillonnage des données plus performante (Class Weight) . Nous avons également créé une métrique métier et fixé un seuil de solvabilité optimum. Le modèle final est un LightGBM optimisé sur la métrique Fbreta2 score et ROC_AUC.

Enfin, voici certaines pistes pour améliorer les résultats et répondre au mieux aux attentes et besoins des conseillers clients :

- Feature engineering en créant des variables plus pertinentes
- Ensembling : moyenner les performances de plusieurs modèles
- Feature selection : supprimer les variables les moins importantes
- Optimisation plus précise des hyperparamètres
- Graphes interactifs pour simplifier l'utilisation de l'interface
- Modification de la métrique créée, avec l'aide d'un expert métier