

Prêt à dépenser

Note Méthodologique

Implémentez un modèle de Scoring

Erwan Berthaud

Prêt à dépenser

DASHBOARD

Client information / Loan request info

Client data

ID Client

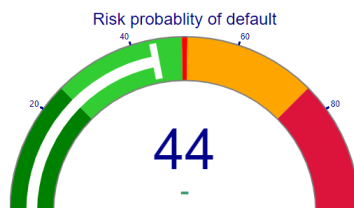
Select a client :

100001.0

SK_ID_CURR	AGE	GENDER	FAMILY STATUS	NB OF CHILDREN	EDUCATION	INCOME SOURCE	YEARS EMPLOYED	INCOME
100,001	53	F	Married	0	Higher education	Working	6	135000

SK_ID_CURR	CONTRACT TYPE	AMOUNT REQUESTED (\$)	ANNUITY (\$)	GOODS' PRICE (\$)	HOUSING TYPE
100,001	Cash loans	568800	20560	450000	House / apartment

Probability of default



Probability of default: MIDDLE LOW

Credit request accorded

Projet 7 du parcours data science de Openclassrooms

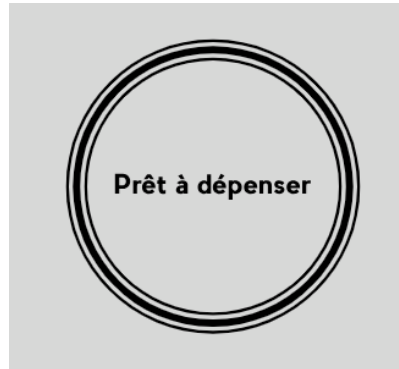
Juin 2023

1. INTRODUCTION	1
2. LES DONNÉES	1
3. TRAITEMENT DES DONNÉES.....	3
4. MODÉLISATION.....	5
5. SYNTHÈSE DES RÉSULTATS.....	9
6. INTERPRÉTABILITÉ DU MODÈLE.....	10
7. ANALYSE DU DATA DRIFT	11
8. LIMITES ET AMÉLIORATION	12
9. TABLEAU DE BORD	12

1. Introduction



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 notation 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 pour entraîner et mettre en production un algorithme de classification supervisée.

Le développement d'un tableau de bord 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. Les données

Huit fichiers au format csv sont disponibles. Ils contiennent 218 informations bancaires et personnelles anonymisées pour 307511 clients. Ces informations ont été recueillies auprès de Home Crédit Group et d'autres institutions bancaires.

Par ailleurs, un fichier test avec 48744 clients nous est fourni, afin de prédire la note de crédit et la probabilité de défaut de ces clients à partir du modèle entraîné.

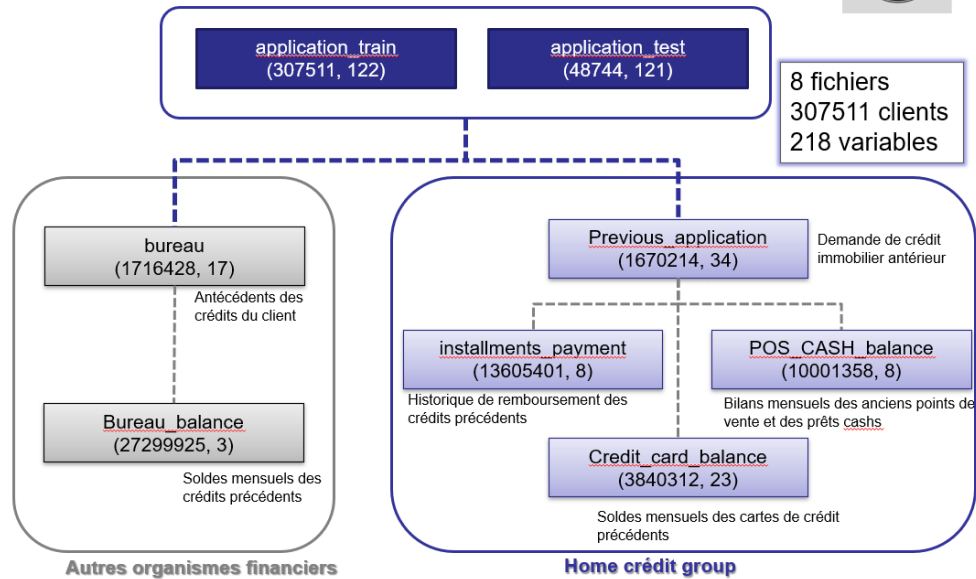


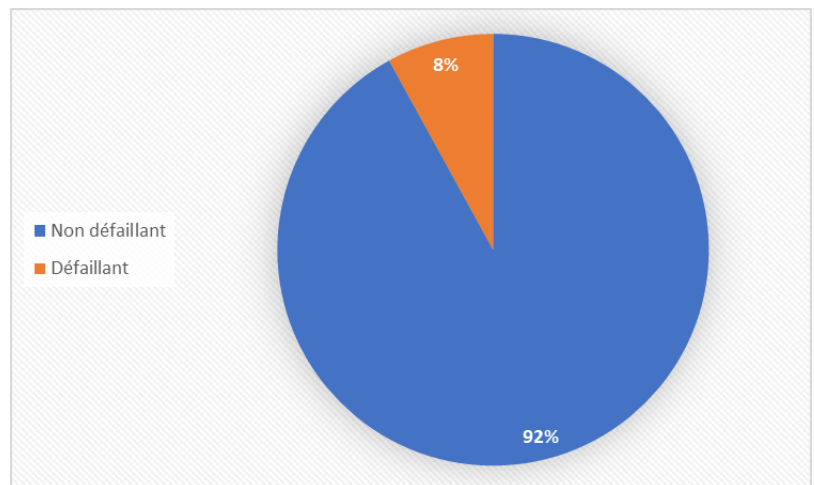
Figure 2-1 : Arborescence du jeu de données

Problématique :

La variable à prédire est une variable binaire :

- La **classe 0** indique un client non défaillant, qui rembourse son crédit
- La **classe 1** indique un client défaillant, qui ne rembourse pas son crédit.

L'objectif est de prédire la classe en calculant la probabilité de défaut. Certains algorithmes sont sensibles aux données déséquilibrées. Une attention particulière au rééquilibrage est portée par la suite.



Pour notre problématique, nous devons minimiser les pertes d'argent pour notre société financière, notre modèle doit donc :

- Ne surtout pas prédire un client non-défaillant s'il est défaillant → minimiser les faux négatifs (prédit non-défaillant mais client défaillant dans la réalité). Dans ce cas, le groupe Home Crédit aura perdu toute la somme prêtée à l'emprunteur. Cela constitue les plus grosses pertes pour l'entreprise.
- S'efforcer de ne pas prédire en défaillant un client non défaillant → minimiser les faux positifs (client prédit défaillant alors que non-défaillant dans la réalité). Dans ce cas, le groupe Home Crédit aura seulement perdu les intérêts de la somme qu'il aurait prêtée à l'emprunteur.

3. Traitement des données

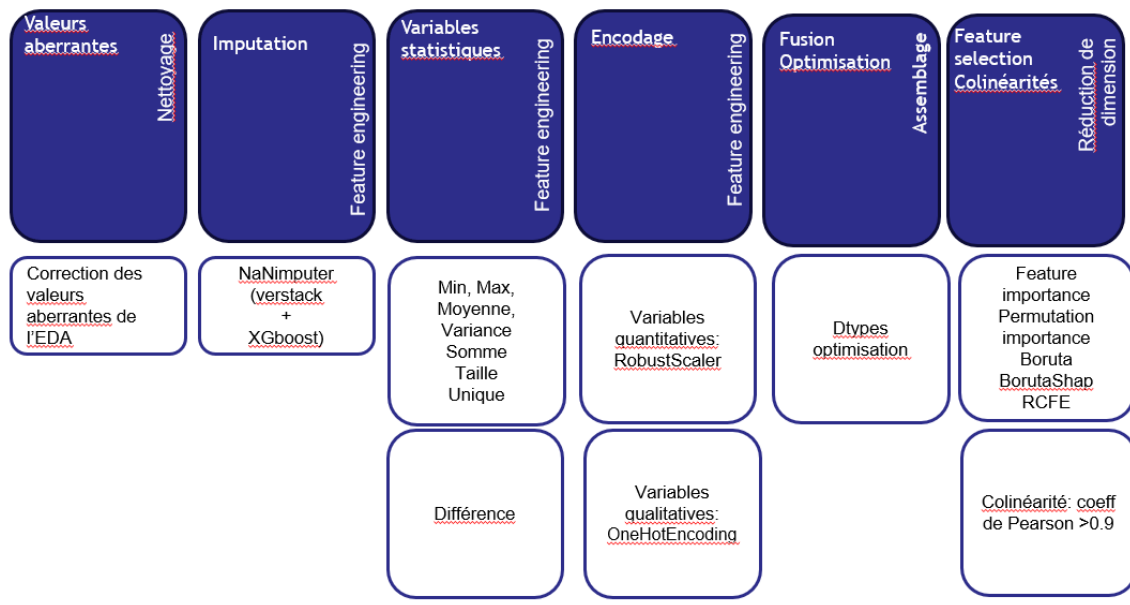


Figure 3-1: processus de traitement des données

Étape 0 : Analyse exploratoire

Cette étape se base sur le kernel kaggle :

<https://www.kaggle.com/code/codename007/home-credit-complete-eda-feature-importance>

Étape 1 : Nettoyage de données

Cette étape consiste à nettoyer et à corriger ou éliminer les valeurs aberrantes.

Étape 2 : Imputation

Les données manquantes, à l'exception de la cible à prédire, sont imputées en utilisant le module de NaNImputer de la bibliothèque Verstack. Cet algorithme utilise XGboost et automatise la pipeline pour entraîner et prédire les valeurs manquantes.

Étape 3 : Feature engineering

Cette étape consiste à créer des nouvelles variables qui peuvent augmenter la performance de prédiction du modèle. Cette étape utilise le kernel kaggle :

<https://www.kaggle.com/code/jsaguiar/lightgbm-with-simple-features>

Les nouvelles variables sont créées en utilisant des statistiques assez simples : Minimum, maximum, moyenne, ratio, variance, size (taille), nunique (nombre de données de la catégorie).

Ce kernel a montré son efficacité dans la compétition et son utilisation est assez simple et permet de réduire le temps passé sur cette étape, qui peut être longue et fastidieuse.

Étape 4: Encodage

Les données sont ensuite encodées :

- Les données numériques sont standardisées en utilisant RobustScaler, qui supprime la médiane et met à l'échelle les données en fonction de la plage des quantiles. Le centrage et la mise à l'échelle

se produisent indépendamment sur chaque fonctionnalité en calculant les statistiques pertinentes sur les échantillons de l'ensemble d'apprentissage.

- Les données catégorielles sont, elles, transformées en utilisant le OneHotEncoding. Exemple : pour une donnée à trois catégories : rouge, vert, bleu, cela va créer 3 nouveaux vecteurs rouge_cat, vert_cat, bleu_cat, qui contiendra des zéros, et un 1 si la variable correspond à la nouvelle catégorie.

Étape 5 : Optimisation

Les données sont ensuite fusionnées en un seul fichier puis optimisées à l'aide de l'optimizer de Verstack. Le fichier contient alors 546 variables. Ce nombre élevé de variable risque de pénaliser les performances de prédiction (qualité et temps d'exécution) du modèle.

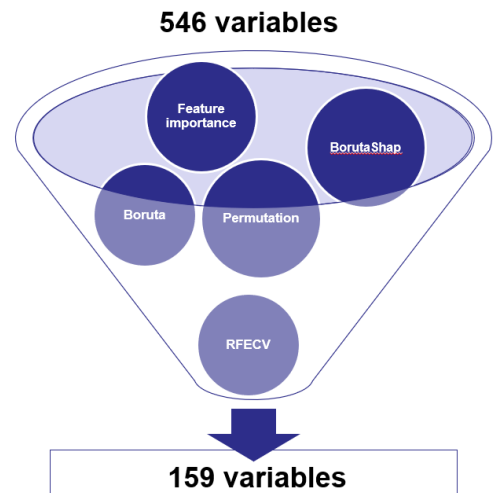
Étape 6 : Réduction de dimension

La sélection de variables consiste, étant donné des données dans un espace de grande dimension, à trouver un sous-ensemble de variables pertinentes. Il faut donc minimiser la perte d'information venant de la suppression de toutes les autres variables.

Plusieurs techniques ont été utilisées :

- Filtrage : suppression des variables colinéaires.
- Embedded : des méthodes intégrées qui apprennent quelles variables contribuent le mieux à la précision du modèle pendant sa création. Une valeur est calculée et liée à chaque variable du jeu de données servant à entraîner le modèle.
- Automatiques : basées sur des bibliothèques python (Boruta, BorutaShap, RFECV, permutation importance avec scikit-learn)

Le fichier final contient 159 variables.

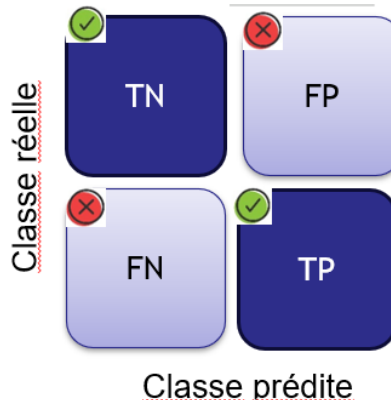


4. Modélisation

Métriques d'évaluation

Le choix de la métrique est primordial, dépend de la problématique et permet d'évaluer la performance du modèle prédictif et de garantir la qualité du modèle de classification. Ces métriques permettent de comparer les classes réelles aux classes prédites par le modèle.

Pour notre problématique, nous devons minimiser les pertes d'argent pour notre société financière.



Les métriques d'évaluation prises en compte dans ce projet sont les suivantes :

- AUC-ROC: Il mesure la capacité d'un modèle à classer correctement les exemples positifs par rapport aux exemples négatifs, quelle que soit la valeur du seuil de classification. L'AUC-ROC varie entre 0 et 1, où une valeur de 1 représente une performance parfaite et une valeur de 0.5 représente une performance aléatoire.
- Accuracy : C'est un score d'exactitude calculé en divisant le nombre d'exemples correctement classés par le nombre total d'exemples. Une valeur de 1 indique que tous les exemples ont été classés correctement et une valeur de 0 indique une classification complètement incorrecte.
- Recall : ou sensibilité. Il mesure la capacité d'un modèle à identifier correctement les exemples positifs (client à risque) parmi tous les exemples réellement positifs présents dans l'ensemble de données. Une valeur de 1 indique un rappel parfait, c'est-à-dire que tous les clients à risque ont été correctement identifiés, et une valeur de 0 indique que aucun client à risque n'a été correctement identifié.
- Precision : Il mesure la capacité d'un modèle à classer correctement les exemples positifs parmi tous les exemples prédits comme positifs. Une valeur de 1 indique une précision parfaite, c'est-à-dire que tous les clients prédits à risque sont corrects, et une valeur de 0 indique que aucun client prédit à risque n'est correct.
- F-bêta : Le score F-bêta est la moyenne harmonique pondérée de la précision et du rappel, atteignant sa valeur optimale à 1 et sa pire valeur à 0. Une valeur bêta plus petite, comme 0.5, accorde plus d'importance à la précision et moins au recall, tandis qu'une valeur bêta plus grande, comme 10, accorde moins d'importance à la précision et davantage au rappel dans le calcul du score.
- Score métier : Il s'agit d'un score créé pour prendre en compte la problématique métier suivante : le coût d'un faux négatif FN (clients à risque prédit fiable : donc crédit accordé et perte en capital) est dix fois supérieur au coût d'un faux positif FP :

$$1 - \frac{FP + 10FN}{N + 10P}$$

Choix du modèle

6

Pour se faire une première idée des modèles de classification performants, le jeu de données a été entraîné en automatique sur tous les modèles de classification de la librairie Pycaret. Pour cette analyse, le rééquilibrage est effectué en utilisant SMOTE, directement avec le pré-processing de Pycaret.

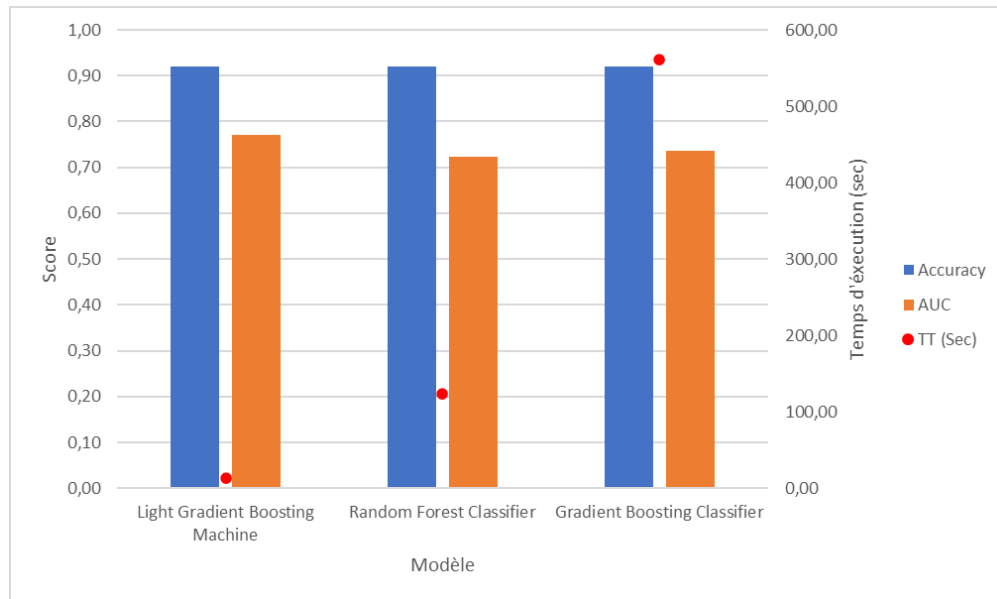


Figure 4-1: Meilleurs modèles sans optimisation - Top 3

Les résultats montrent que les modèles (LGBM, RandomForest et Gradient Boosting) semblent être plus performants sur notre jeu de données.

Le modèle LightBGM non optimisé est très rapide et obtient des résultats satisfaisants qui pourront être améliorés (optimisation par réglage des hyperparamètres) et réglés (réglage de la valeur de seuil minimal de probabilité pour faire basculer la prédiction vers les classes positives = client défaillant) pour maximiser les métriques adéquates, c'est ce modèle qui sera retenu pour être optimisé.

Traitement du déséquilibre

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).

Deux méthodes principales existent pour égaliser les classes :

- le sur-échantillonnage (Oversampling)
- et le sous-échantillonnage (Undersampling).

3 méthodes ont été testées :

- SMOTE (Synthetic Minority Oversampling Technique)
- l'hyperparamètre `class_weight` du modèle LightGBM.
- l'hyperparamètre `scale_pos_weight` du modèle LightGBM.

 SMOTE	 Class weight	 Scale_pos_weight
Créer des données synthétiques de la classe minoritaire	Créer un modèle qui attribue des poids différents pour chaque classe pour pénaliser la classe majoritaire	Modifie le seuil de classification de probabilité de la classe en favorisant une des classes

Le rééquilibrage avec l'hyperparamètre `class_weight` est celui qui donne les meilleurs résultats.

Optimisation du modèle



La technique retenue pour l'optimisation des hyperparamètres du modèle LightGBM est l'optimisation Bayésienne avec 2 librairies différentes (skopt de scikit-learn et optuna). Cette optimisation est effectuée directement avec Pycaret, ce qui permet d'automatiser la pipeline et d'éviter la fuite de données.

L'optimisation bayésienne fonctionne en construisant une distribution postérieure de fonctions (processus gaussien) qui décrit au mieux la fonction que l'on veut optimiser. Au fur et à mesure que le nombre d'observations augmente, la distribution postérieure s'améliore, et l'algorithme devient plus certain des régions de l'espace des paramètres qui méritent d'être explorées et de celles qui ne le méritent pas.

L'optimisation a été effectuée sur les deux différentes métriques (F-bêta et score métier) pour le jeu de données traités.

Réal	0	43653	12884
	1	1785	3180
		0	1
		Prédiction	

Figure 4-2: Matrice de confusion du modèle LGBM optimisé

Dans un second temps, le seuil de classification, 0.5 par défaut, est optimisé pour les deux métriques F-bêta et score métier.

Le but est de minimiser le nombre de faux négatifs tout en prédisant le plus de vrais positifs possibles tout en limitant le nombre de faux positifs. Le modèle LightGBM avec les paramètres de base sert de comparatif.

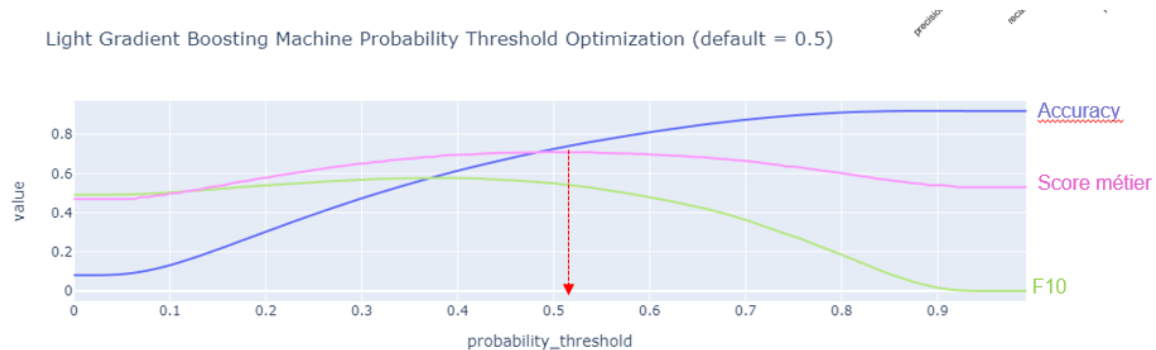


Figure 4-3: Courbe des scores en fonction du seuil de probabilité de classement - Optimal à 0.53

5. Synthèse des résultats



le tableau ci-dessous synthétise les résultats initiaux et ceux du modèle optimisé.

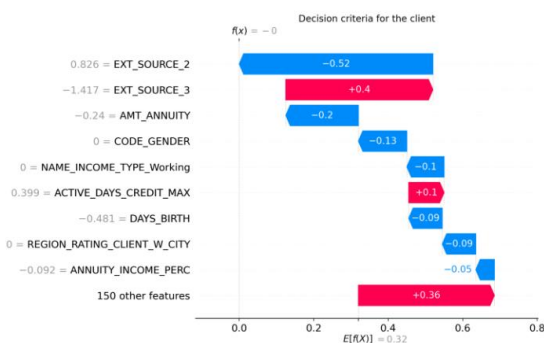
Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	F10	Score Métier	TT (Sec)
Light Gradient Boosting Machine optimisée	0,76	0,78	0,64	0,20	0,30	0,20	0,26	0,53	0,71	12,43
Light Gradient Boosting Machine	0,92	0,77	0,02	0,53	0,04	0,04	0,10	0,02	0,54	13,59
Random Forest Classifier	0,92	0,72	0,00	0,53	0,00	0,00	0,03	0,00	0,53	123,68
Gradient Boosting Classifier	0,92	0,74	0,01	0,50	0,01	0,01	0,05	0,01	0,54	561,49
Extra Trees Classifier	0,92	0,72	0,00	0,54	0,01	0,00	0,03	0,00	0,53	67,24
Dummy Classifier	0,92	0,50	0,00	0,00	0,00	0,00	0,00	0,00	0,53	1,97
Extreme Gradient Boosting	0,92	0,77	0,05	0,46	0,10	0,08	0,13	0,06	0,55	358,59
Ada Boost Classifier	0,91	0,70	0,05	0,24	0,08	0,05	0,07	0,05	0,55	106,05
Decision Tree Classifier	0,84	0,54	0,17	0,14	0,15	0,07	0,07	0,17	0,56	29,86
K Neighbors Classifier	0,74	0,58	0,34	0,12	0,17	0,06	0,07	0,29	0,57	491,98
Linear Discriminant Analysis	0,70	0,76	0,69	0,17	0,27	0,16	0,22	0,54	0,70	14,01
Ridge Classifier	0,70	0,00	0,69	0,17	0,27	0,16	0,22	0,54	0,70	3,44
Logistic Regression	0,64	0,62	0,54	0,12	0,20	0,08	0,11	0,41	0,60	73,89
SVM - Linear Kernel	0,59	0,00	0,59	0,11	0,19	0,06	0,10	0,42	0,59	4,52
Quadratic Discriminant Analysis	0,36	0,62	0,79	0,09	0,17	0,03	0,07	0,47	0,54	11,73
Naive Bayes	0,15	0,57	0,94	0,08	0,15	0,00	0,02	0,48	0,48	9,69

6. Interprétabilité du modèle

Les valeurs SHAP sont utilisées pour évaluer l'importance de chaque variable dans la décision d'octroi de crédit pour un client donné. Les variables ayant une influence négative sur l'accord du crédit sont mises en évidence en rouge, tandis que celles ayant une influence positive sont en bleu. Cette distinction permet de mieux comprendre les facteurs qui défavorisent ou favorisent l'octroi du crédit. L'analyse des variables en rouge identifie les éléments ayant une influence défavorable, utile pour d'éventuels ajustements ou améliorations. De même, les variables en bleu indiquent les éléments favorisant l'accord du crédit. Cette visualisation combinée avec les valeurs SHAP offre une interprétation plus complète et intuitive de la contribution de chaque variable à la décision d'octroi de crédit pour le client donné.

Cette visualisation peut être réalisée à l'échelle globale du modèle, figure de droite, ou à l'échelle locale, c'est-à-dire d'un client, figure de gauche.

Client:



Global:

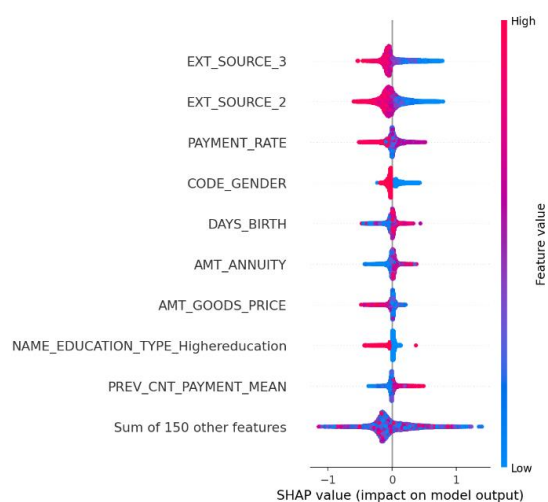


Figure 6-1: Influence des variables dans la prédiction - client prédit non-défaillant

On retrouve 3 types de variable :

- Les données externes : EXT_SOURCE_1 et EXT_SOURCE_2
- Les informations bancaires : ratio annuité du prêt sur le montant du crédit, Durée du crédit précédent à l'application de la demande précédente
- Des informations personnelles : date de naissance, sexe, éducation.

7. Analyse du data drift



La dérive des données fait référence à un phénomène dans lequel les caractéristiques ou les distributions des données changent au fil du temps. Cela peut se produire dans les ensembles de données utilisés pour l'apprentissage automatique lorsque les données d'entraînement et les données d'évaluation ne sont pas cohérentes.

La dérive des données peut se produire pour diverses raisons, telles que des changements dans l'environnement, des changements dans le comportement des utilisateurs ou des problèmes dans le processus de collecte des données. Elle peut entraîner une baisse des performances des modèles prédictifs, car ces derniers sont formés sur des données qui ne sont plus représentatives des données actuelles.

La détection et la gestion de la dérive des données sont des tâches importantes dans le domaine de l'apprentissage automatique. Cela peut impliquer la surveillance régulière des performances du modèle, la collecte continue de nouvelles données, l'adaptation du modèle aux changements et la réévaluation périodique du modèle pour maintenir sa précision et sa fiabilité dans des conditions changeantes.

Une analyse du Data Drift en production a été réalisée en utilisant la bibliothèque evidently. Cette bibliothèque permet de détecter d'éventuels changements de données (Data Drift) sur les principales caractéristiques entre les données d'entraînement et les données de test. Une synthèse est automatiquement générée sous la forme d'un tableau HTML d'analyse.

En comparant les distributions des 120 variables du jeu d'entraînement et du jeu de test, un drift a été détecté sur 9 variables, soit 7.5% du total. Le seuil de détection du data drift est fixé de 50%, il n'y a pas de dérive de données entre le jeu d'entraînement et le jeu de test.

Dataset Drift		
Dataset Drift is NOT detected. Dataset drift detection threshold is 0.5		
120	9	0.075
Columns	Drifted Columns	Share of Drifted Columns
Data Drift Summary		
Drift is detected for 7.5% of columns (9 out of 120).		

8. Limites et amélioration

12

Pour répondre au problème de classification binaire à partir des 8 fichiers fournis par Prêt à dépenser, de nombreuses techniques de Machine Learning ont été nécessaires : rééquilibrage des classes, création de nouvelles variables facilement explicables, sélection des variables pour rendre le modèle moins complexe, choix des métriques adaptées à notre problématique métier, réflexion sur le compromis taux de faux négatifs et taux de faux positifs et sur le réglage du seuil de décision.

Néanmoins une collaboration avec notre client permettrait d'améliorer le modèle. Les experts métiers pourraient nous aider à créer une métrique bancaire plus efficace et adaptée et pourraient nous donner leur avis sur l'intérêt des nouvelles variables créées et pourquoi pas nous indiquer de nouvelles variables. Une explication des données externe serait un plus puisqu'il est difficile d'être transparent en utilisant ces variables importantes pour le modèle mais inexplicables pour le client.

Le Dashboard pourra également être évalué par le client et les remarques prises en compte, des améliorations techniques (cache mémoire des fonctions statiques, taille des graphiques affichés, couleurs, chartre graphique du client ?) pourraient également être améliorées.

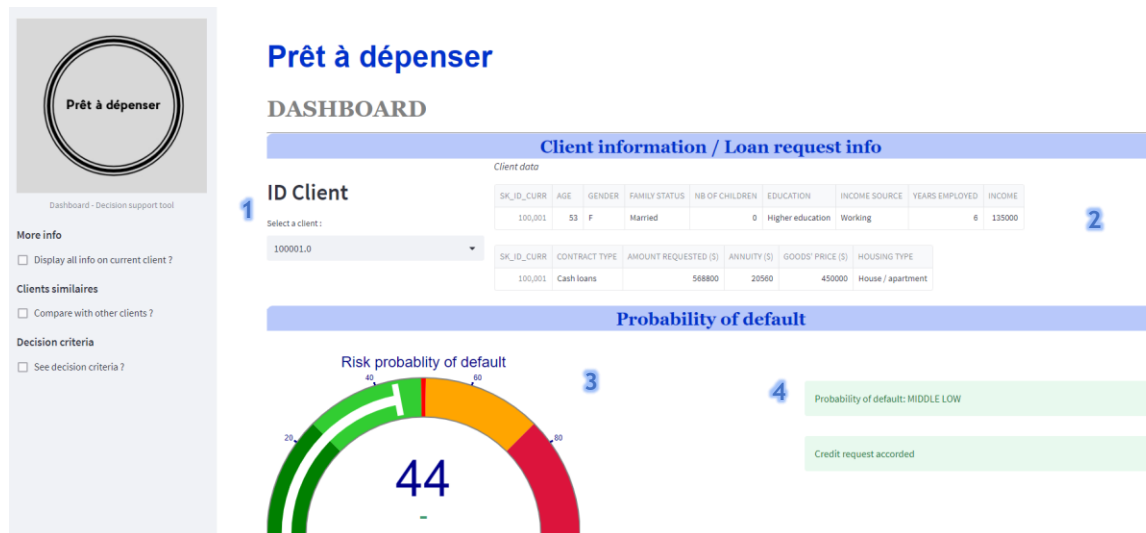
Enfin, une discussion serait nécessaire pour déterminer quelles variables sont réellement exploitables sans poser de questions de confidentialités ou d'éthiques (le sexe, par exemple, est un critère discutable).

9. Tableau de bord

Dépôt des sources : <https://github.com/sefirotha/OC-DS-P7>

Dépôt MLflow : https://dagshub.com/sefirotha/OC-DS-P7_mlflow/experiments/#!/

Tableau de bord : <https://credit-score-eb-e593e2243d0a.herokuapp.com/>



1. Choix du client
2. Informations sur le client
3. Score du client
4. Décision d'accord ou non du crédit
5. Afficher toutes les informations du client
6. Voir où le client se situe par rapport aux autres clients
7. Voir les facteurs influençant la décision finale pour le client