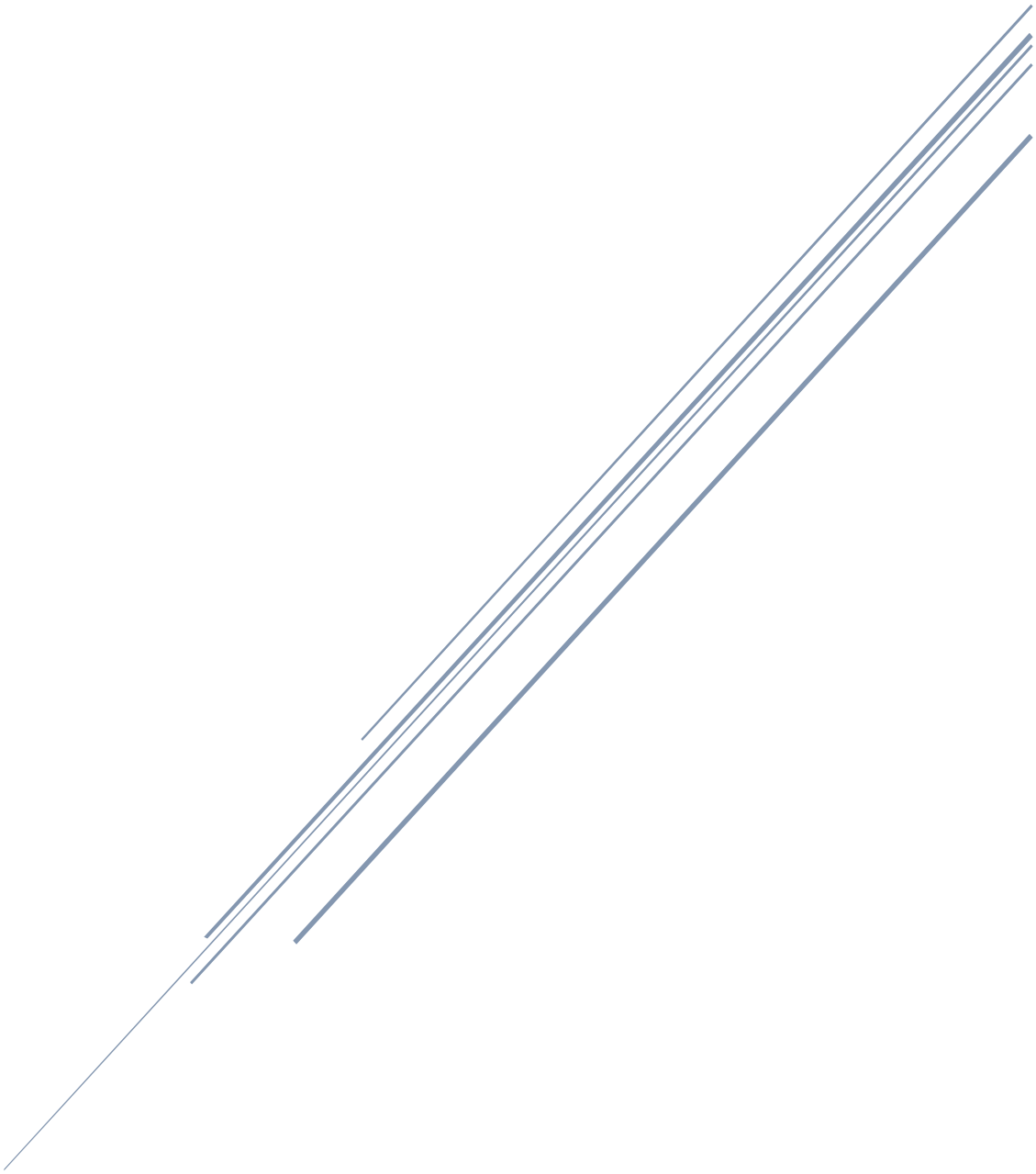
IMPLEMENTEZ UN MODELE DE SCORING

Projet 7 – Parcours Data Scientist



Open Classrooms
Franck Le Mat

Table des matières

Contexte

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 pour appuyer la décision d'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.). Cet outil sera accompagné d'un dashboard interactif permettant d'interpréter les prédictions faites par l'outil.

Prédire si un client remboursera ou non un prêt ou s'il rencontrera des difficultés à le faire est un besoin commercial essentiel. C'est un problème nécessitant l'utilisation d'un modèle de classification binaire supervisé. Les données sont fortement disproportionnées avec plus de 280000 clients ayant remboursé leur prêt et un peu moins de 25000 ayant fait défaut.

Traitement des données

La base de données de « Home Credit » comporte initialement plus de 300 000 clients ainsi que leurs données personnelles sur leur demande de prêt, leur historique de remboursement, leur historique de précédents crédits à « Home Credit » et dans d'autres établissements bancaires. Ces données sont réparties dans sept jeux de données, plus un fichier de description des variables. Il est possible de joindre les données entre elles grâce aux identifiants clients ou les identifiants des demandes de prêts antérieures (Figure1).

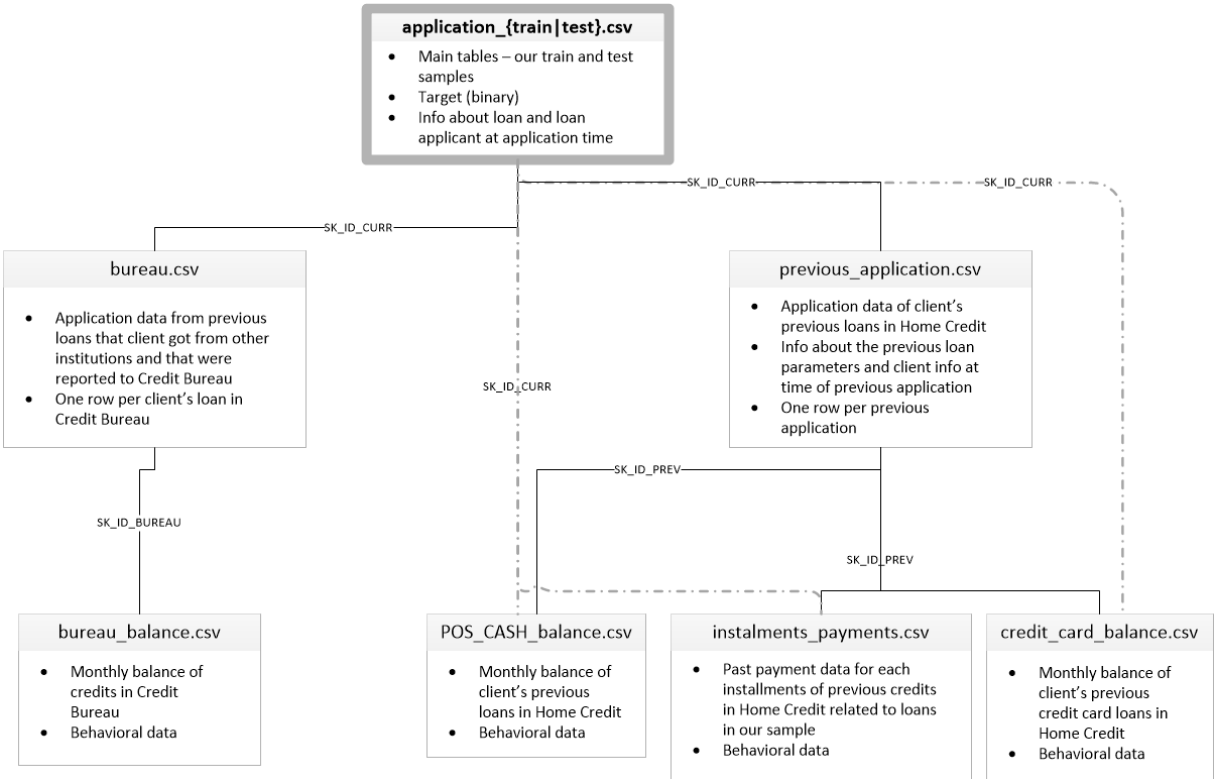


Figure 1. Structure du jeu de données. Source : https://www.kaggle.com/c/home-credit-default-risk/data

Dans un premier temps, une analyse exploratoire a permis de déterminer les variables ayant un impact significatif sur la probabilité d'un client de rembourser ou non leur prêt (target). Cette évaluation a été faite à l'aide de test statistiques ANOVA pour les variables continues et test de Cramer's V pour les variables catégorielles.

Afin d'obtenir le jeu de données final, c'est-à-dire un seul fichier avec une ligne par client, une première étape d'agrégation a été nécessaire sur les jeux de données comportant plusieurs entrées pour un même client (i.e un client peut avoir plusieurs prêts antérieurs à agréger en une seule ligne). Cette agrégation est donc à l'origine de transformation de variable. Les variables catégorielles ont été transformer avec un One Hot Encoder puis la moyenne de chaque possibilité a été calculé. Aussi, pour les variables continues de nouvelles variables agrégées ont été calculées comme le minimum, le maximum, la moyenne, la somme et l'écart-type. Seul le jeu de données nommée « application_train » n'a pas eu à subir d'agrégation.

Ensuite à partir de ce jeu de données final, les valeurs manquantes ont été gérées par suppression des variables avec un taux de remplissage inférieur à 60 % sauf si cette dernière faisait partie des variables les plus fortement corrélé à la variable target (ANOVA).

Une étude approfondie de la corrélation entre les variables (matrices de Spearman) a permis de supprimer de nombreuses variables trop corrélées entre elles (si coefficient de corrélation supérieur ou égal à 0,9).

A la fin de cette étape de pré-traitement le jeu de données final comportait 431 variables dont 9 catégorielles, 421 continues et la variables target.

Modélisation

La figure 2 montre la méthodologie employée pour trouver le meilleur modèle de classification.

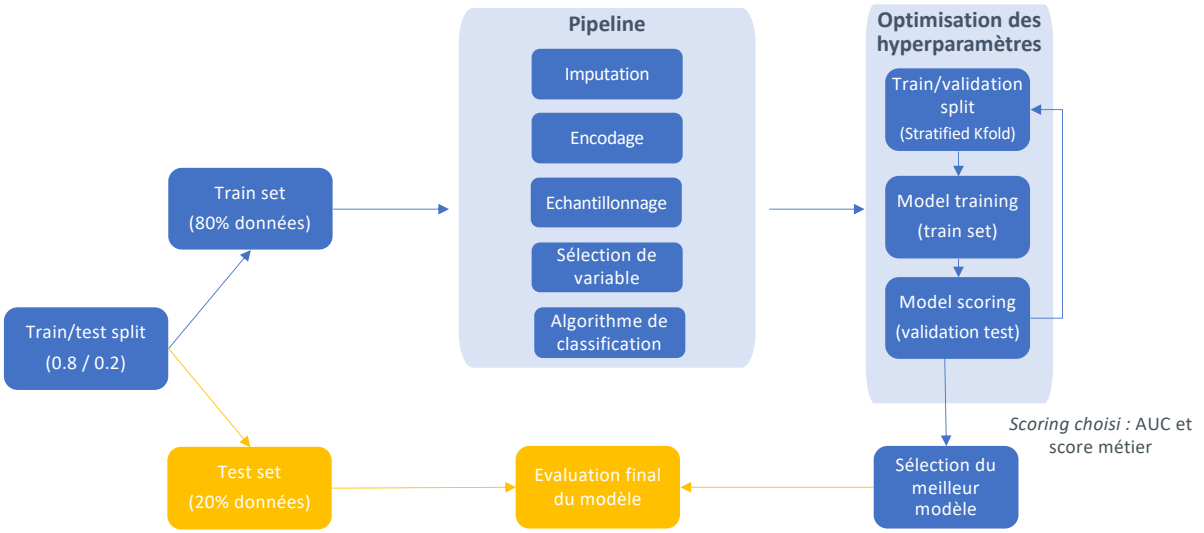


Figure 2 : Organigramme de la méthodologie d'évaluation du meilleur modèle de classification.

Entraînement du modèle

Il a été nécessaire d'évaluer plusieurs étapes de pré-traitement. L'imputation qui représente la l'imputation ou non des valeurs manquantes restantes ou simplement la suppression totale de toutes les valeurs manquantes.

L'encodage qui est la transformation des variables explicatives : les variables catégorielles transformées avec un One Hot Encoder et les variables numériques centrées/réduites avec un Standard Scaler.

Comme mentionné précédemment, dans les données il y a un fort déséquilibre entre les clients ne présentant pas de risque de défaut de payement et ceux présentant un risque. Ce déséquilibre va avoir un impact sur la performance de l'algorithme et les prédictions seront faussées car le modèle attribuera beaucoup plus fréquemment la classe la plus représentée aux clients. Pour pallier ce déséquilibre, quatre approches peuvent être appliquées sur les données d'entraînement :

• Class weights est une méthode directement gérée par les modèles et qui permet de pénaliser les poids associés aux observations de la classe sur-représentées (ici solvable).

• Over-sampling est une méthode qui va dupliquer aléatoirement des données existantes de la classe sous-représentée pour que chaque classe ait le même nombre de données que la classe sur-représentée à l'origine.

• SMOTE est une méthode qui va créer de nouvelles données pour la classe sous-représentée à partir des données existantes (et donc de la variété) pour que chaque classe ait le même nombre de données que la classe sur-représentée à l'origine.

• Under-sampling est une méthode qui va sélectionner une partie des observations sur-représentées pour que chaque classe ait le même nombre de données que la classe sous-représentée à l'origine.

Le temps d'entrainement d'un modèle est aussi un facteur de performance. Ayant avant prétraitement plus de 400 variables explicatives pour plus de 300 000 clients une étape de sélection de variable afin de diminuer un peu le temps de traitement a été ajouté dans la phase de prétraitement.

Enfin, pour déterminer l'algorithme optimal de classification adapté à la problématique, quatre algorithmes ont été testés. Un algorithme dit naïf, comme baseline, qui prédit uniquement la classe la plus représentée (pour nous pas de risque de défaut de payement), un algorithme de régression binomiale (régression logistique), un algorithme de forêt d'arbre aléatoire (Random Forest Classifier) et un algorithme de gradient boosting (Light-GBM).

Choix de la métrique d'évaluation

Métriques d'évaluation

Une matrice de confusion (Figure 3) est l'un des meilleurs moyens d'évaluer les performances d'un modèle de classification. L'idée de base est de compter le nombre de fois que les instances d'une classe sont correctement classées ou incorrectement classées comme une autre classe. Les lignes d'une matrice de confusion représentent la classe réelle, tandis que chaque colonne représente la classe prédite. Un modèle parfait n'aurait que de vrais positifs et de vrais négatifs, ce qui signifie que sa matrice de confusion n'aurait que des valeurs non nulles sur sa diagonale. Dans une matrice de confusion, il existe deux types d'erreurs : les faux positifs (FP) ou les erreurs de type I et les faux négatifs (FN) ou les erreurs de type II. Ces termes proviennent de tests d'hypothèses en statistique et sont utilisés de manière interchangeable avec les problèmes de classification.

		Classe réelle	
		-	+
Classe prédite	-	True Negatives *(vrais négatifs)*	False Negatives *(faux négatifs)*
	+	False Positives *(faux positifs)*	True Positives *(vrais positifs)*

Figure 3 : Matrice de confusion des prédictions d'un modèle de classification binaire.

Pour notre problématique, le résultat attendu le plus important pour un client est la valeur de la probabilité de défaut de payement. En appliquant un seuil à cette valeur, nous pouvons lui affecter une valeur binaire (0 ou 1) suivant que la probabilité est inférieure ou supérieure au seuil.

Si la probabilité est inférieure au seuil, on considère que le crédit sera remboursé, la prédiction est négative (0). Inversement, si la probabilité est supérieure au seuil, on considère que le crédit ne sera pas remboursé, la prédiction est positive (1).

Ainsi :

- Accorder un crédit à un client ne pouvant pas le rembourser par la suite (FN) est synonyme de perte.
- Accorder un crédit à un client qui le remboursera par la suite (TN) est un gain.
- Ne pas accorder le prêt et que le client ne peut pas rembourser (TP) n'est ni une perte, ni un gain.

- Ne pas accorder le prêt alors que le client pouvait rembourser (FP) est une perte de client donc d'argent.

Ces valeurs peuvent être traduites en des indicateurs caractérisant le modèle, dont voici les plus importants :

- Sensibilité = TP / (TP + FN) - Capacité du modèle à détecter les dossiers de crédit non remboursés (1)
- Spécificité = TN / (TN + FP) - Capacité du modèle à détecter les dossiers de crédit remboursé (0)
- Précision = TP / (TP + FP) - Capacité du modèle à détecter les vrais dossiers non remboursés (1)

Nous pouvons représenter les évolutions de la sensibilité et de la spécificité en fonction du seuil de classification avec une courbe ROC (Figure 4).

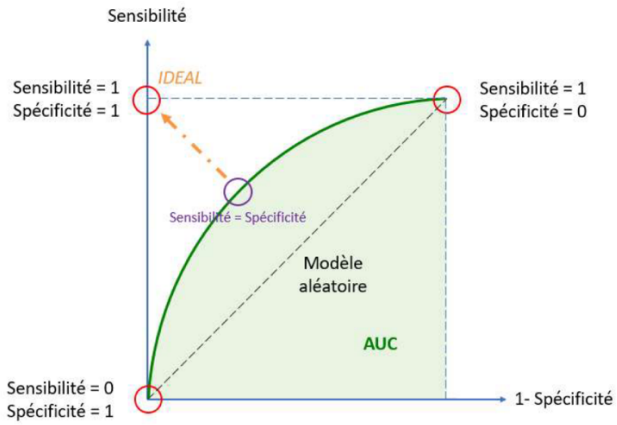


Figure 4 : Courbe ROC d'un modèle de classification binaire.

Une courbe ROC caractérise le classifieur qui a produit les résultats sous forme de probabilités par variation du seuil de classification. Un modèle idéal a une sensibilité et une spécificité = 1. Plus la courbe se rapproche de cet idéal, meilleurs sont les indicateurs.

⇒ On peut résumer la mesure de cette performance par l'aire sous la courbe (Area Under the Curve - AUC).

Ainsi pour le modèle utilisé pour cette étude, nous avons cherché à maximiser le score AUC.

Fonction coût (métrique métier)

Optimiser la valeur de la mesure AUC permet d'améliorer globalement la sensibilité et la spécificité. Cette approche est pertinente si on considère les éléments de la matrice de confusion de même importance.

Dans le domaine bancaire, un crédit non remboursé coûte plus cher qu'un dossier de crédit non signé. Il s'agit de trouver le meilleur compromis entre le nombre de crédit qu'on accorde mais qui ne seront in fine pas remboursés (les faux négatifs) et le nombre de crédit qu'on refuse et dont on perd potentiellement le bénéfice sur les intérêts pour les clients solvables (les faux positifs).

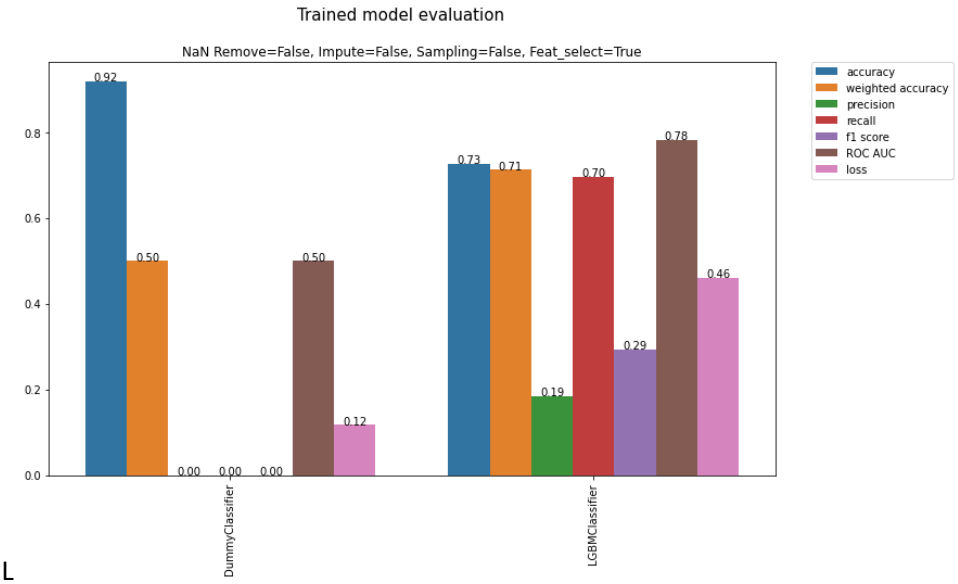
Une société de crédit cherche à maximiser ses gains. Il est alors possible de créer une fonction coût qui sera représentatif de ces gains par rapport à ces gains maximums potentiel.

Pour cela nous avons besoin de connaître le maximum de bénéfice possible. En utilisant les valeurs de la matrice de confusion nous pouvons calculer le nombre total de client qui sont capable de rembourser le prêt soit $TN + FP$. Soit α le bénéfice moyen si un client a remboursé sont prêt. Les gains maximums sont donc de $\alpha \times (TN + FP)$. Cependant, les réels bénéfices fait par la société sont les gains obtenus par le remboursement d'un prête plus les pertes subit par le défaut de remboursement. En prenant cette fois β comme la perte moyenne si un client fait défaut les gains sont $\alpha \times TN + \beta \times FN$. Donc la fonction coût est : $\frac{\alpha \times TN + \beta \times FN}{\alpha \times (TN+FP)}$

Sans information métier concrète nous avons utilisé des valeurs estimées pour le coefficient alpha et beta, 20 000 et 200 000 respectivement. Cette fonction coût a été implémentée afin de pénaliser l'impact des erreurs sur la décision d'octroi de crédit.

Résultat de l'évaluation du pré-traitement

Les résultats des différents tests sur pré-traitement ont montré que l'algorithme **lightGBM** était le plus pertinent pour notre modélisation à la fois en termes de performance de prédiction qu'en temps de d'entraînement. Les étapes de prétraitement suivant donnaient les meilleurs résultats : **conservation des valeurs manquantes, encodage des variables, gestion de la target déséquilibré à l'aide de la « Class weights » de l'algorithme et sélection de variables pour la rapidité d'exécution.** La Figure 5 montre les résultats sur les mesures les plus communes de performance de modèle en comparaison avec un Dummy Classifier (les algorithmes RandomForest et Logistic Rgression ne fonctionnant pas avec des valeurs manquantes, ils n'ont pas pu être testé sur cette condition). La variables « loss » corresponds à la fonction coût développé ci-dessus.



L

Figure 5 : Résultat sur différente mesure de lightGBM vs Dummy Classifier.

Optimisation Bayésienne

L'optimisation bayésienne est un algorithme d'optimisation basé sur l'apprentissage automatique utilisé pour trouver les paramètres qui optimisent globalement une fonction de boîte noire donnée. Il y a 2 composants importants dans cet algorithme :

- La fonction boîte noire à optimiser : $f(x)$. On veut trouver la valeur de x qui optimise globalement f(x). Le $f(x)$ est aussi parfois appelé la fonction « objectif », la fonction cible ou la fonction de perte selon le problème. En général, nous ne connaissons que les entrées et les sorties de $f(x)$.
- La fonction d'acquisition : a(x), qui est utilisée pour générer de nouvelles valeurs de x à évaluer avec $f(x)$. $a(x)$ s'appuie en interne sur un modèle de processus gaussien $m(X, y)$ pour générer de nouvelles valeurs de x.

Le processus d'optimisation lui-même est le suivant :

- Définir la fonction boîte noire $f(x)$, la fonction d'acquisition $a(x)$ et l'espace de recherche du paramètre x.
- Générez aléatoirement des valeurs initiales de x et mesurez les sorties correspondantes à partir de $f(x)$.
- Ajuster un modèle de processus gaussien $m(X, y)$ sur $X = x\ et\ y = f(x)$. En d'autres termes, $m(X, y)$ sert de modèle de substitution pour $f(x)$!
- La fonction d'acquisition $a(x)$ utilise ensuite $m(X, y)$ pour générer de nouvelles valeurs de x comme suit. Utilisez $m(X, y)$ pour prédire comment f(x) varie avec x. La valeur de x qui conduit à la plus grande valeur prédite dans $m(X, y)$ est alors suggérée comme prochain échantillon de x à évaluer avec $f(x)$.
- Répétez le processus d'optimisation aux étapes 3 et 4 jusqu'à ce que nous obtenions finalement une valeur de x qui mène à l'optimum global de $f(x)$. Notez que toutes les valeurs historiques de x et $f(x)$ doivent être utilisées pour former le modèle de processus gaussien $m(X, y)$ dans la prochaine itération - à mesure que le nombre de points de données augmente, $m(X, y)$ devient meilleur pour prédire l'optimum de $f(x)$.

L'optimisation a été faite en utilisant l'AUC et la fonction coût métier comme fonction boîte noire, et ont donnée respectivement des modèles avec un AUC de 0,768 et de fonction coût de 0,471 avec des paramètres très proche pour les modèles optimisés.

Interprétabilité du modèle

Maintenant que le modèle, le pré-traitement et les paramètres de l'algorithme ont été définis, il est désormais intéressant de savoir quelles sont les informations qui ont un poids important dans le calcul de la probabilité de solvabilité d'un client. Le modèle Light-GBM est un modèle basé sur des arbres et l'importance des features est donnée par la fonction feature_importance avec pour option importance_type='gain'. L'importance des features se base sur la réduction moyenne de la perte obtenue lors de l'entraînement du modèle (figure 6).

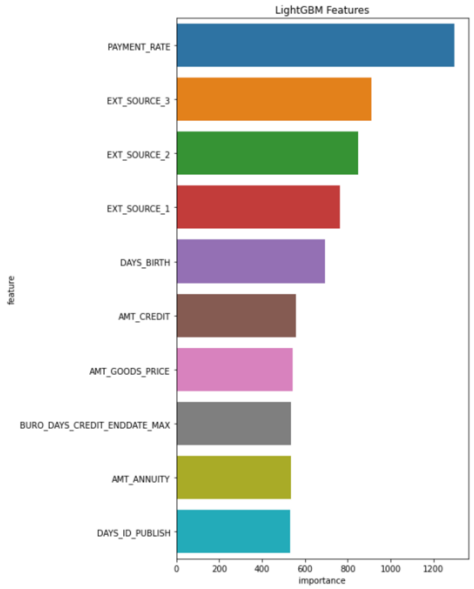


Figure 6 : Représentation des 10 features les plus importantes dans la prédiction.

Avec cette représentation, on peut dire que les features les plus importantes pour la prédiction d'accord d'un prêt sont payement rate, les sources extérieures 1, 2 et 3 qui sont des scores normalisés créés à partir de sources de données externes (cela est d'ailleurs en concordance avec les résultats de l'étude exploratoire). Puis on trouve le nombre de jours depuis la naissance des clients donc leur âge joue un rôle important dans l'acceptation d'un crédit, ainsi que le montant des annuités.

Pour plus de précision et pour connaître le sens d'influence de chacune des variables localement pour chaque prédiction, il est possible d'utiliser la librairie SHAP (SHapley Additive exPlanations) qui explique la sortie de tout modèle d'apprentissage automatique en utilisant la théorie des jeux (Figure 7).

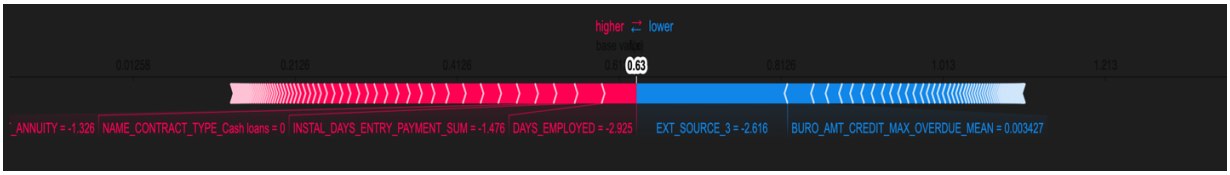


Figure 7 : SHAP force plot d'influence des variables sur la prédiction (La couleur bleu indique les caractéristiques qui poussent la prédiction plus haut, et la couleur rose indique exactement le contraire..)

Limite

La principale limite actuelle du modèle est qu'il ne comporte pas de traitement spécifique des valeurs manquantes. Les variables d'entrée sont donc toutes indispensables à son bon fonctionnement.

Le feature engineering a été effectué sans connaissances réelles du secteur bancaire. Les nouvelles variables créées restent des transformations basiques des variables de base (moyenne, minimum, maximum, écart-type et quelques taux). Aussi, la modélisation a été effectuée sur la base d'une métrique généraliste AUC ou bien personnellement créée pour répondre au mieux au besoin de gain d'argent d'une banque. Les coefficients de cette métrique ont été choisis arbitrairement selon le bon sens. L'axe principal d'amélioration serait donc de définir plus précisément ces coefficients associés à chaque combinaison classe prédite/classe réelle car le modèle déterminé ici ne sera pas obligatoirement le meilleur.

La sélection des variables corrélées est ici basée sur le coefficient de corrélation de Spearman, mais une meilleure expertise métier permettrait d'effectuer un tri plus cohérent, notamment en sélectionnant les variables non corrélées qui ont le plus de sens pour les assureurs.

Les axes d'amélioration sont donc :

La mise en place d'une étape de traitement des valeurs manquantes, adaptée aux gros datasets, avec une étude du biais introduit dans les données par une telle méthode (un ré-échantillonage post-traitement pour conserver les distributions du dataset de base pourrait être nécessaire).

Un échange avec les experts métier permettrait :

- D’effectuer un feature engineering plus pertinent, par exemple avec le calcul d'indicateurs spécifiques au monde des assurances,
- De mieux sélectionner les variables corrélées à enlever ou conserver en fonction des connaissances métier,
- De réduire le nombre de variables pour optimiser le temps de calcul en sélectionnant celles qui ont de faibles coefficients dans la régression logistique et un faible intérêt du point de vue métier.