Création d'un modèle de scoring et d'un tableau de bord

Cyril REGAN

14 septembre 2020

Problématique

Affaires :

- Crédits à la consommation

Besoin :

- Modèle de scoring sur : probabilité de faillite d'un client.
- **Tableau de bord** interactif pour interpréter les prédictions.

Données

Données kaggle : **Home Credit Default Risk**.
Utilisation du notebook **a-gentle-introduction** pour pré-traiter les données.

- 307511 clients
- 244 catégories + 1 cibles (remboursement ou non par les clients des prêts)
- 8% des clients en défaut de paiement : déséquilibre de classe important
  => traitement spécifique

Difficultés

Scoring :
Déséquilibre de classe : difficultés des modèles pour prédire la classe minoritaire.

- Techniques de gestion du déséquilibre
- Choix ou construction d'une métrique pertinente

Tableau de bord :

- Trouver et interpréter les caractéristiques importantes pour expliquer la prédiction
- Présenter clairement le score du client et sa position par rapport à un groupe de clients similaires.

Table of Contents

Techniques d'échantillonage

Oversampling SMOTE (Synthetic Minority Over-Sampling TEchnique) : ajoute des données de la classe minoritaire par combinaison avec les voisins proches.

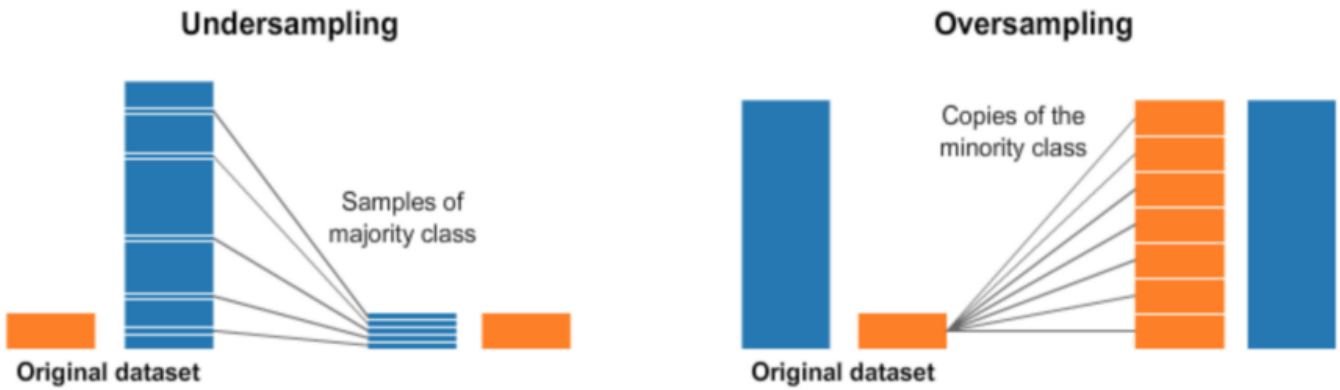


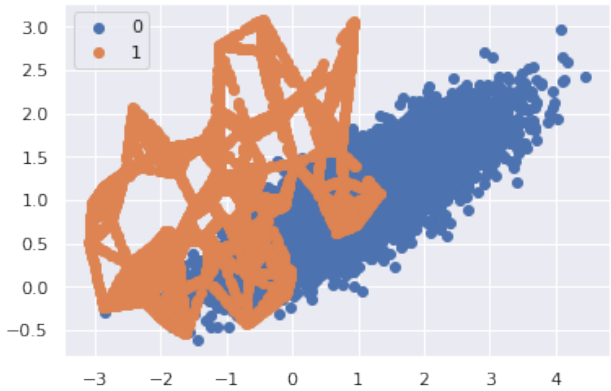
Figure – Techniques d'échantillonage

Undersampling : enlève des données de la classe majoritaire aléatoirement.

Effet de SMOTE et undersampling sur données 2D déséquilibrées :

Initial

SMOTE



Undersampling

SMOTE : génère nx pts entre les pts d'origine (classe minoritaire).

Undersampling : réduction drastique des pts (classe majoritaire).

Poids différentiés (class-weighting)

Class_weight (dans **sklearn**) : attribution de plus de poids à certaines observations.
Class_weight = "balanced" : attribution des poids en fonction de la proportion des classes.

Table of Contents

Classification binaire

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

Matrice de confusion

- *Justesse*(*accuracy*) : proportion de prédiction exacte.
- $Rappel = \frac{TP}{TP+FN}$: taux de vrais positifs. Proportion de positifs que l'on a correctement identifiés.
- $Precision = \frac{TP}{TP+FP}$: proportion de prédictions correctes parmi les points que l'on a prédits positifs.

- $F - mesure = 2 \times \frac{Precision \times Rappel}{Precision + Rappel}$: moyenne harmonique entre rappel et précision.
- $Specificite = \frac{TN}{FP+TN}$: taux de vrais négatifs.

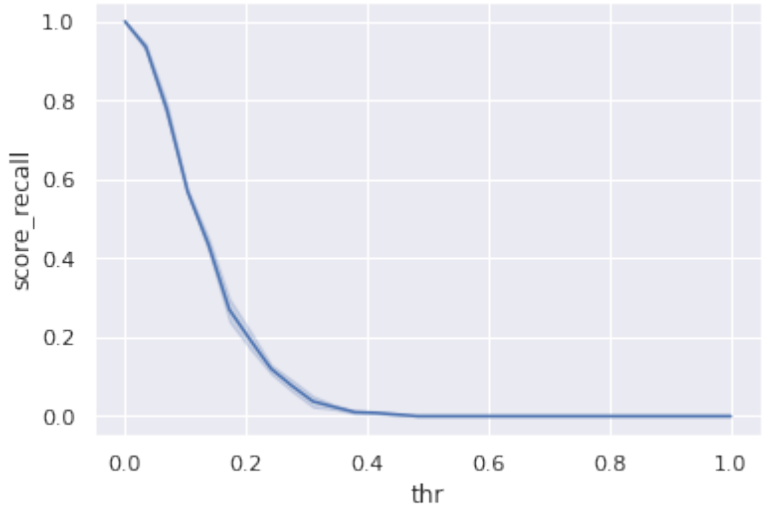
Que de métriques !
Mais sont-elles toutes pertinentes pour notre étude ?

- ~~Justesse~~ : un modèle qui prédit que des négatifs aura une justesse 92% ! Bon score = bon modèle ? 😐... non !

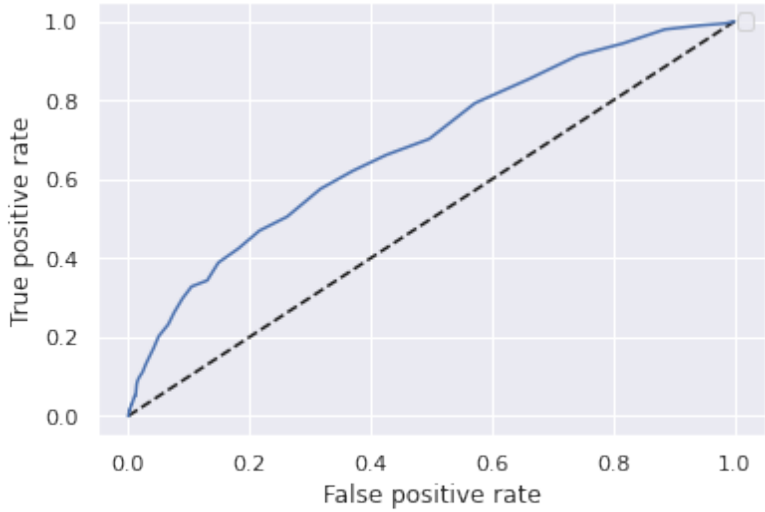
=> Rappel : plus adapté car il détecte l'erreur sur les clients n'ayant pas recouvert leur prêt 👍 !

Score

La plupart des algos retournent en fait un score (probabilité qu'un point est positif) sur lequel un seuil (thr) est défini pour une prédiction binaire. Par défaut $thr = \mathbf{0.5}$ mais optimisation possible :



RF : $Rappel(thr)$



RF : ROC Curve $\frac{Rappel}{1-Specificite}(thr)$

L'AUROC est l'aire sous la courbe ROC.

Métrique spécifique

En entrée : matrice de confusion (TN, FN, TP, FP)
Calcul : principe de gain et de perte :

$$G = \sum_{i \in TN} C[i] \tag{1}$$

Gain (G) = somme des montants des crédits (C) des clients prédits solvables ayant recouvert leur prêts (vrais négatifs : TN)

$$G_{max} = \sum_{i \in N} C[i] \tag{2}$$

Gain max (G_{max}) = somme des montants des crédits sur tous les clients ayant recouvert leur prêts (négatifs : N)

$$L = \sum_{i \in FN} AMT\_CREDIT[i] \tag{3}$$

Perte (L) = somme des montants des crédits des clients prédits solvables n'ayant pas recouvert leur prêts (faux négatifs : FN).

$$\text{score} = (G - 10.L)/G_{max} \tag{4}$$

=> Le **score** vaut le gain - 10 fois les pertes, normalisé sur le gain maximal.

Table of Contents

Séparation jeux de données

- Jeu d'entraînement : 4/5.
- Jeu de test : 1/5.
- Même répartition de classe pour les 2 jeux.

Validation croisée

Vigilance sur technique d'échantillonage : Transformation **uniquement sur le jeu d'entraînement** sans modification du jeu de test.

En validation croisée : vigilance assurée pour chaque pli (fold)

=> Construction de sa propre validation croisée (sans utiliser **GridSearch de Sklearn**).

Table of Contents

Évaluation de la gestion du déséquilibre

Évaluation de l'under-oversampling ou class-weighting par :

- Modèle : foret aléatoire.
- Métriques classiques : justesse (accuracy), f-mesure (f1), précision (precision), rappel ou sensibilité (recall), l'AUROC (aire sous la courbe ROC).
- Validation croisée sans optimisation d'hyper-paramètres.

Optimisation métrique spécifique :

- Technique de gestion du déséquilibre (la plus performante).
- Modèle : Light Gradient Boosting Machine (LightGBM).
- Validation croisée avec hyper-paramètres :

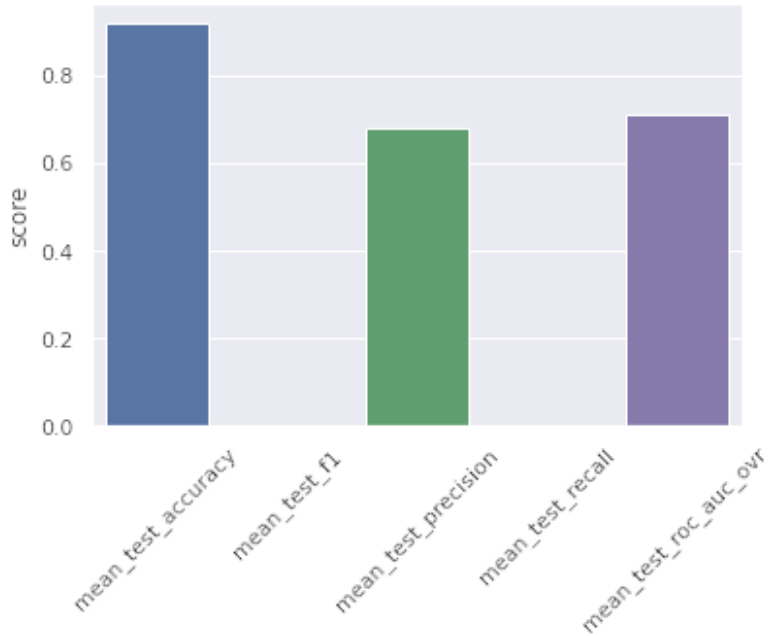
"n_estim"	[100, 1000]	"max_depth"	[5,7,-1]	"num_leaves"	[9,31,127]
"objective"	binary	"class_weight"	balanced	"learning_rate"	[0.03,0.05]
"reg_alpha"	0.1	"reg_lambda"	0.1	"subsample"	0.8
"n_jobs"	-1	"random_state"	50		

- Variation du seuil : $thr \in [0...1]$

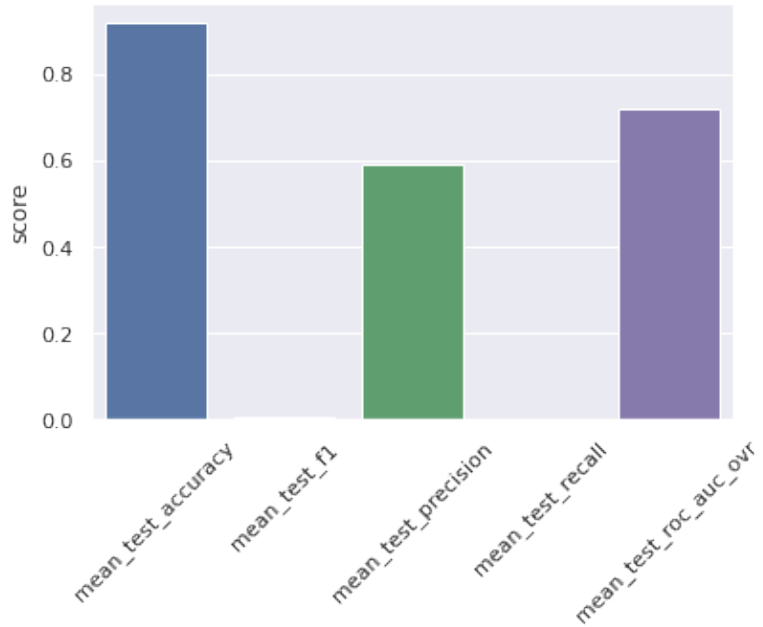
Table of Contents

Méthodes de gestion du déséquilibre

Scores avec déséquilibre

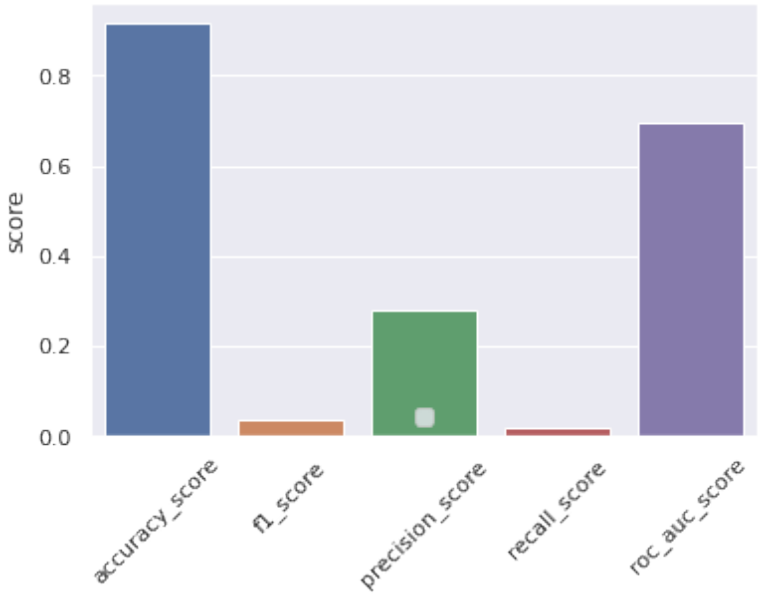


Scores avec class-weighting

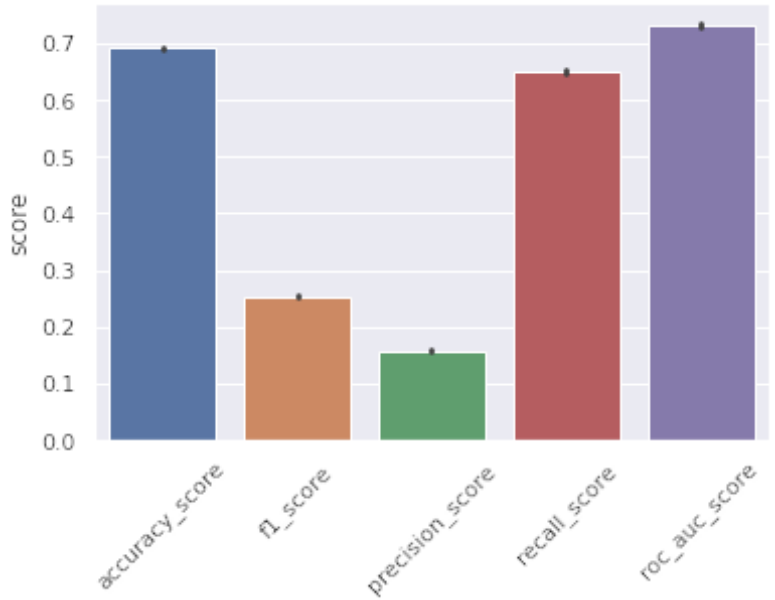


Le class-weighting n'améliore pas les résultats.

Scores avec oversampling



Scores avec undersampling



Oversampling SMOTE : pas d'amélioration significative.

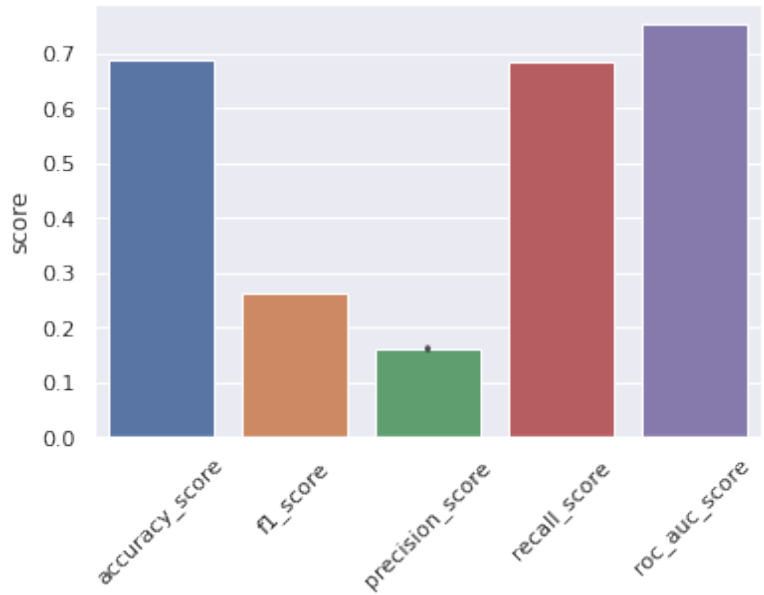
Undersampling : plus précis pour le **rappel** (critère le plus important pour évaluer le risque de défaut)

=> **Undersampling** retenu pour optimisation de métrique spécifique.

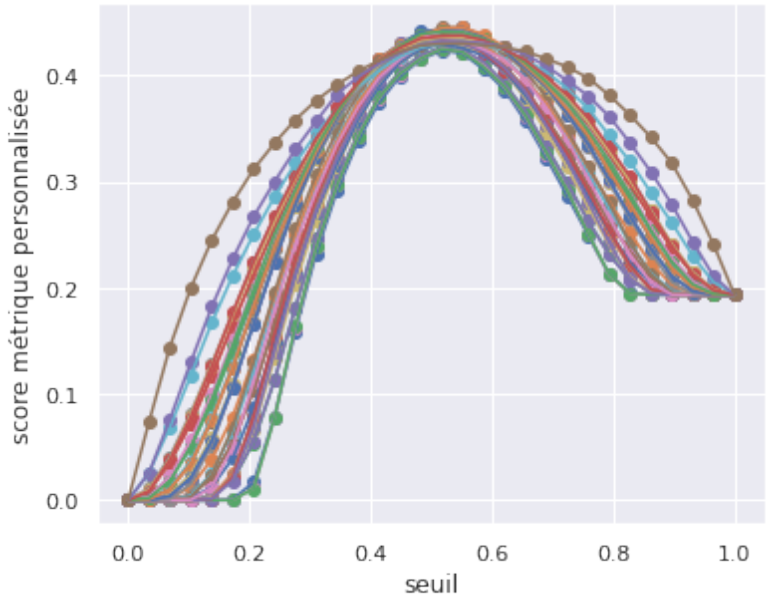
Table of Contents

Optimisation métrique spécifique

LightGBM : métriques classiques [folds + params]



LightGBM : métrique spec [folds + params](seuil)



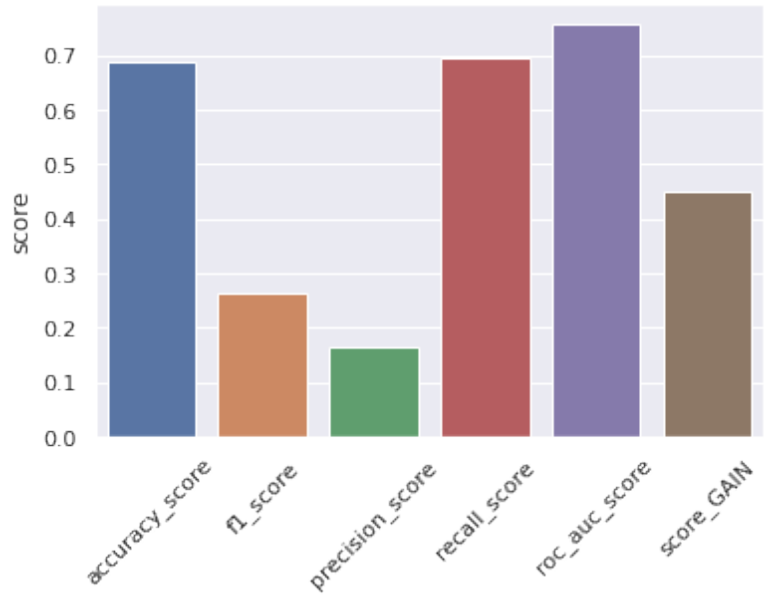
Scores métriques classiques LightGBM : meilleurs que RF (rappel = 0.69 / 0.67 pour RF)

Score optimisé métrique spécifique : **0.44** pour un seuil de **0.52**

Table of Contents

Validation test

Modèle final : métriques classiques + spécifique :
Résultats similaires à ceux de l'entraînement



=> Validation modèle final.

Table of Contents

Interprétation

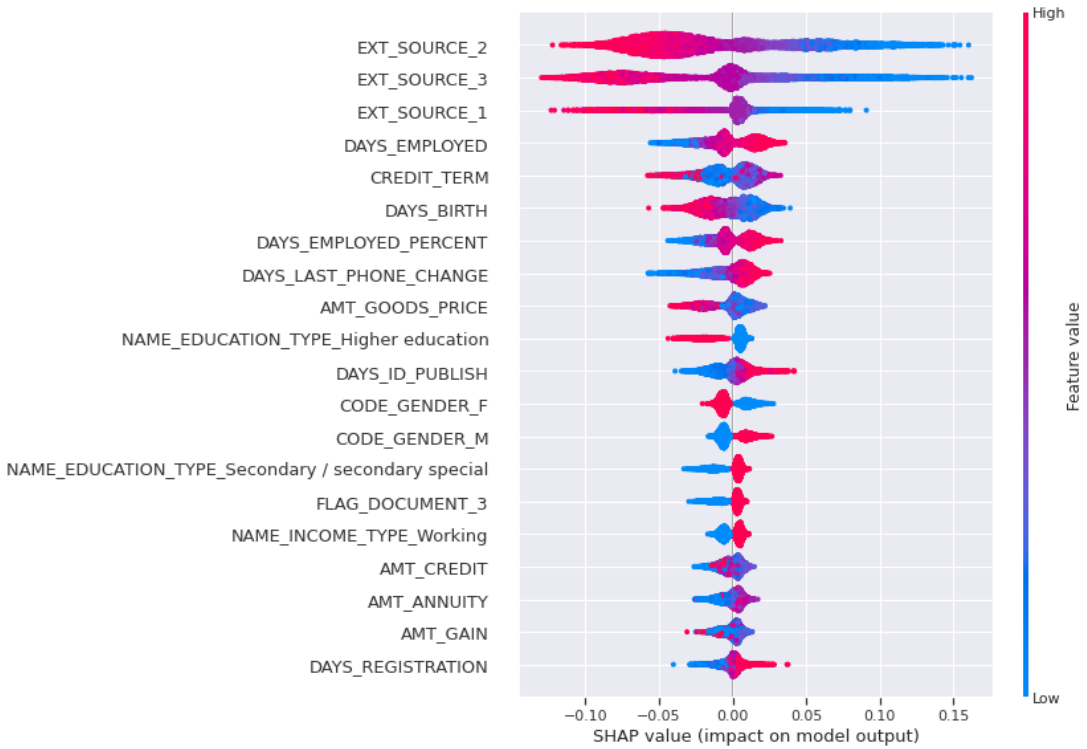
Éthique et RGPD (Règlement Général sur la Protection des Données) : nécessité d'expliquer la prédiction d'une intelligence artificielle (IA).
Méthodes :

- Importance des variables : utile mais uniquement de manière globale.
- LIME (Local Interpretable Model-Agnostic Explanations) : observe l'influence des variables sur des individus particuliers. Bien mais limitée par le voisinage des observations (problématique avec classes déséquilibrées)
- SHAP (Local SHapley Additive exPlanations) influence des variables sur individus particuliers sans limites par voisinage (basée sur théorie des jeux).

Les valeurs SHAP permettent donc :

- de définir l'**importance des caractéristiques**
- sur une **partie** de la population,
- et d'**interpréter l'influence** (positive ou négative) des caractéristiques sur la prédiction (*Risk*)

Valeurs SHAP sur modèle final :



3 plus importantes caractéristiques : EXT_SOURCE_1, 2, 3

Influence négative sur prédiction : EXT_SOURCE_2 $\nearrow \Rightarrow$ *Risk* $\searrow$

Influence positive sur prédiction : DAY_EMPLO.. $\nearrow$ 0 $\Rightarrow$ *Risk* $\nearrow$

Tableau de bord

Fonctionnement :

- Affiche la densité de toute la population sur les caractéristiques importantes (EXT_S_1, 2, 3, DAY_Employed, Credit_Term, DAY_Birth)
- Sélection du client :
  - Affiche la prédiction (métrique spécifique) du client
  - Affiche position du client par rapport aux 500 plus proches clients voisins sur les caractéristiques importantes

Techno : Flask + React + Heroku : **dashboard**

Conclusion

- Comparaison (RF) techniques gestion déséquilibre de classe : class-weighting, oversampling et **undersampling**
- Choix métrique classique et construction métrique spécifique
- Optimisation (et validation sur jeu de test) modèle LightGBM + métrique spécifique (gain et de perte)
- Détermination de l'importance des caractéristiques + interprétation valeurs SHAP :
  - EXT_SOURCE_2 $\searrow \Rightarrow$ *Risk* $\nearrow$
  - DAY_EMPLOYED $\nearrow 0 \Rightarrow$ *Risk* $\nearrow$
- Déploiement **tableau de bord** Flask + react + heroku

Perspectives

- Prédiction du modèle à améliorer par plus de feature-engineering (cf **notebook kaggle**)
- Entraînement et optimisation d'un XGboost (plus précis) pour améliorer les résultats.
- Construction d'une métrique plus adaptée (somme des intérêts cumulés du prêt pour les gains)

Merci de votre attention !