

# CONSTRUISEZ UN MODÈLE DE SCORING

Hourdin Charlène

Juin 2022



# AGENDA



**Présentation du  
projet**



**Préparation des  
données**



**Modélisation  
et optimisation du  
modèle**



**Interprétabilité du  
modèle (SHAP)**



# Présentation du projet



# Construisez un modèle de scoring

- |    |                        |   |
|----|------------------------|---|
| 01 | LE PROJET              | Mettre en œuvre un outil de “scoring crédit” qui calcule la probabilité qu’un client le rembourse ou non, puis classifie la demande : crédit accordé ou refusé. |
| 02 | LES OBJECTIFS          | Développer un algorithme de classification pour aider à décider si un prêt peut être accordé à un client.   |
| 03 | LA MISSION             | Réaliser une première exploration et visualisation des données  |
| 04 | LA MÉTHODE             | Créer un modèle facilement interprétable avec une mesure de l’importance des variables  |
| 05 | LES RÉSULTATS ATTENDUS | Un Notebook Jupyter, présentant une analyse complète, pertinente et présentable.  |

# Jeu de données

*Diagramme expliquant comment les fichiers CSV sont liés entre eux :*

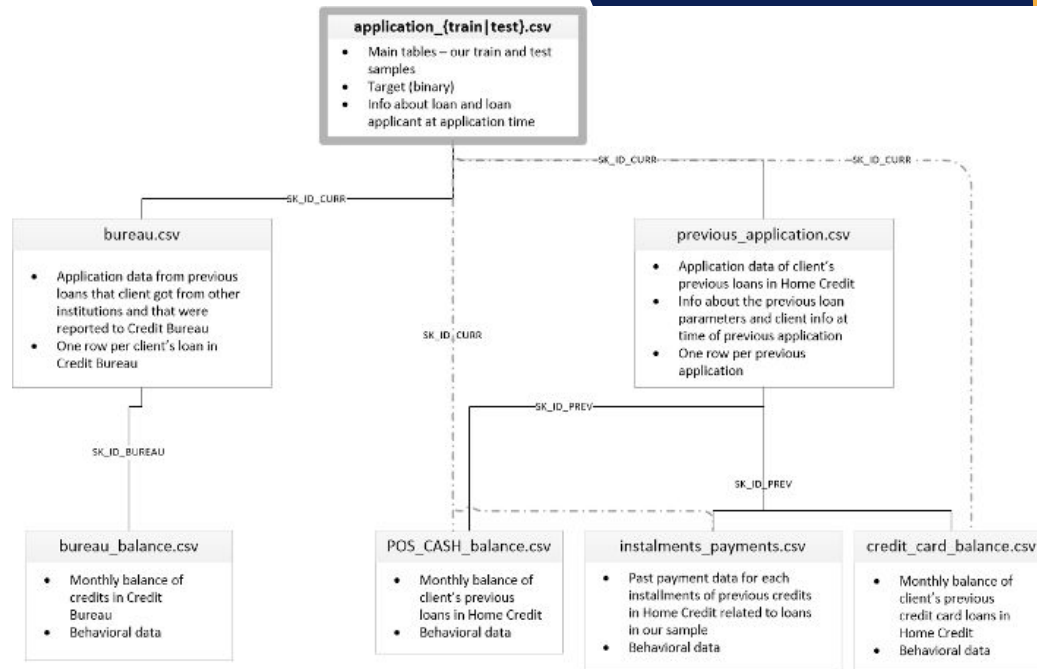
Nous avons à notre disposition un jeu de données qui contient :

- Un historique de prêts
- Un historique d'informations financières
- Des informations sur le comportement des emprunteurs (si il a fait défaut ou pas)

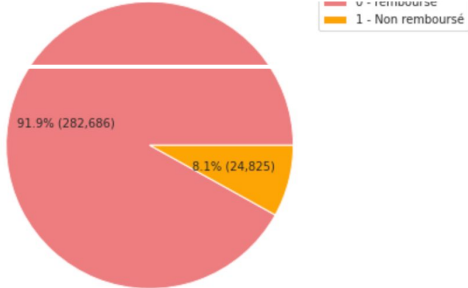
Il y a 7 fichiers CSV

Pour le fichier **application\_train** Il y a au total **307 511** prêts de renseignements, ainsi que **122 features** dont la colonne **TARGET** indiquant si le client n'a pas remboursé le prêt (**TARGET=1**) ou si il l'a fait (**TARGET=0**)

Le fichier **application\_test** contient quand à lui **48744 prêts**, mais pas de colonne TARGET



# Analyse exploratoire

Base de données principale (application_train)	Classe déséquilibré	Implémentation d'un modèle de classification
307.500 clients	<p>Distribution de la variable TARGET</p>  <p>La variable <b>TARGET</b> est déséquilibrée (environ 92% de <b>TARGET=0</b>)</p>	Classification supervisée
121 variables (features)		Variable à prédire : <b>TARGET</b>
<b>Données personnelles du demandeur:</b>  sexe, âge au moment de la demande, situation familiale, nombre d'enfants, profession, revenus, niveau d'études, informations sur le lieu de résidence, fournitures (voiture, maison, téléphone, etc.);		<b>Classification binaire :</b> <b>0 = Client solvable</b> <b>1 = Client non solvable</b>
<b>Données liées au crédit :</b>  identifiant unique, type de crédit, montant du crédit, montant de la rente, documents fournis, etc		<b>La sortie :</b> Probabilité de défaut de paiement
<b>Informations externes (scores) :</b>  Scores achetés à des institutions financières externes.	Utilisation de technique de <b>rééquilibrage des données</b> pour ne pas introduire un biais lors de l'entraînement de notre modèle.  (SMOTE, class_weight)	<b>Apprentissage supervisée :</b>  Cela consiste à superviser l'apprentissage de la machine en lui montrant des exemples (des données) de la tâche qu'elle doit réaliser.

# Préparation des données



# Étape et préparation des données

CLEANING	TRAITEMENT DES ANOMALIES	<ul style="list-style-type: none"><li>- Suppression des étiquettes non présentes sur le jeu de données TEST</li><li>- Remplacement des valeurs extrêmes par des valeurs NaN</li></ul>
CLEANING	TRAITEMENT DES VALEURS MANQUANTES	<ul style="list-style-type: none"><li>- Suppression des colonnes ayant plus de 60 % de valeurs manquantes</li><li>- Imputation des valeurs manquantes par la médiane</li></ul>
FEATURE ENGINEERING	ENCODAGE DES VARIABLES CATÉGORIELLES	<ul style="list-style-type: none"><li>- Utilisation de <b>LabelEncoder()</b> pour les variables binaires</li><li>- Utilisation de <b>OrdinalEncoder()</b> pour la variable sur le type d'étude suivi (classement entre 1 et 5)</li><li>- Pour les autres <b>pd.get_dummies</b> tranforme en colonnes indicatrices (colonnes de 0 et de 1).</li></ul>
FEATURE ENGINEERING	SUPPRESSION ET AJOUT DE VARIABLES (FEATURES)	<ul style="list-style-type: none"><li>- Suppression des variables à faible variance</li><li>- Création de 8 variables pour améliorer le modèle.</li></ul>

Dimension du TRAIN : **307.500, 231** features dont la variable **TARGET**



# Modélisation et optimisation du modèle



# Modélisation et optimisation

## Split / Normalisation

### Train-test-split :

Test size = 20%

### Normalisation :

Mise à l'échelle

## Evaluation des différents modèles

### 7 modèles évalués

#### Baseline :

Dummy Classifier

#### 3 modèles linéaire :

Naïves bayes

SGDClassifier

Régression logistique

#### 3 modèles non-linéaire :

RandomForestClassifier

LightGBM

XGBoost

## Optimisation des hyper-paramètres

### Objectif :

Augmenter les performances du modèle sélectionné en optimisant la métrique

**fbeta\_score**

### GridSearchCV :

Méthode permettant d'évaluer la meilleure combinaison d'hyper-paramètres

## Prédiction et évaluation

### Metricue d'évaluation :

#### AUC :

la probabilité de défaillance  
(comprise entre 0 et 100%)

#### Recall :

Capacité du modèle à détecter tous les clients  
**non-solvable**

#### Fbeta\_score :

capacité du modèle à détecter les VRAIS clients non-solvables

## Interpretabilité

### Approche global :

permet d'expliquer toutes les observations en même temps, globalement.

### Approche locale :

donnent une interprétation pour un seul ou un petit nombre d'observations

# Évaluation des modèles

## Rééquilibrage des classes de la variable cible TARGET :

Utilisation du paramètre :

**Class\_weight = "balanced"**

*(essaie réaliser avec SMOTE mais les résultats étaient moins performant)*

## Cross validation :

Utilisation de l'intégralité du jeu de **TRAIN** pour l'entraînement ET la validation

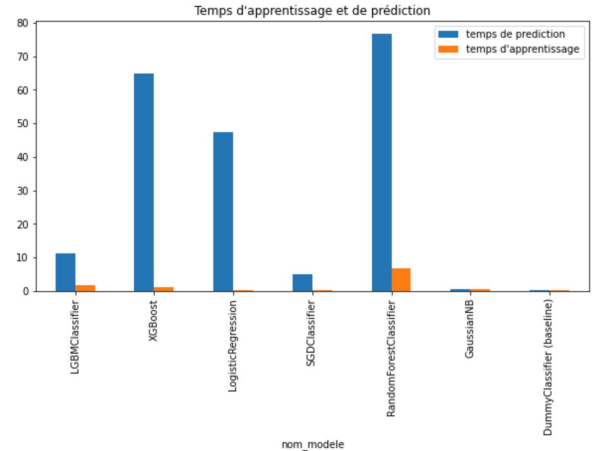
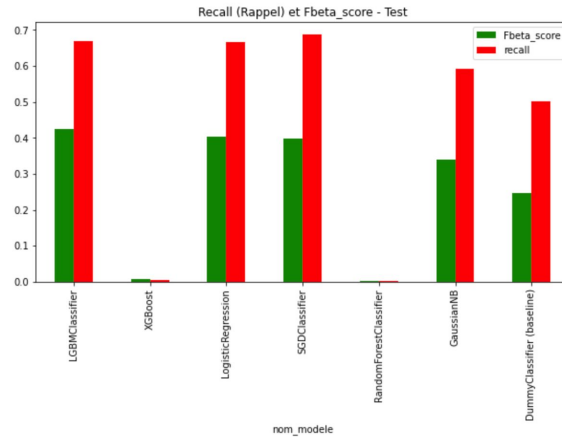
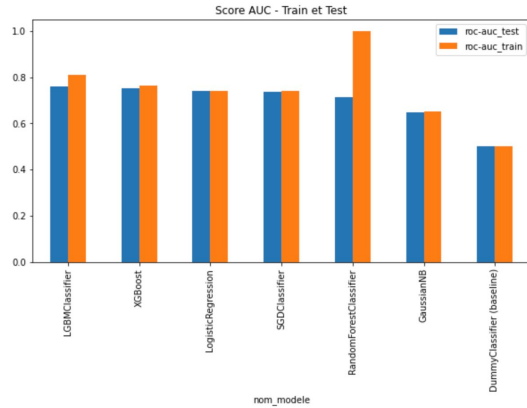
**Choix de 3 folds pour les modèles**

## StratifiedKfold :

Création de sous-ensemble de validation croisée en gardant la même proportion d'exemples pour chaque classe

nom_modele	accuracy_train	accuracy_test	roc-auc_train	roc-auc_test	recall	F1-score	Fbeta_score	temps de prediction	temps d'apprentissage	temps_total
DummyClassifier (baseline)	0.499648	0.499943	0.500000	0.500000	0.502065	0.139486	0.246139	0.202646	0.191264	0.393910
GaussianNB	0.632768	0.633028	0.651141	0.648874	0.593312	0.207000	0.339712	0.623845	0.511521	1.135366
SGDClassifier	0.658551	0.657159	0.739033	0.736293	0.688185	0.245396	0.399038	5.026447	0.215169	5.241616
LogisticRegression	0.684659	0.684049	0.742651	0.740454	0.667288	0.254267	0.404476	47.442389	0.222516	47.664905
RandomForestClassifier	0.999951	0.919394	1.000000	0.712934	0.002316	0.004617	0.002893	76.747968	6.855001	83.602969
LGBMClassifier	0.722319	0.711524	0.809837	0.759501	0.670561	0.272875	0.423611	11.279650	1.751229	13.030879
XGBoost	0.919561	0.919435	0.762708	0.751347	0.005791	0.011468	0.007221	64.729322	1.031811	65.761134

# Comparaison des modèles



**Meilleure performance AUC sur le TEST**  
**Pour le modèle LGBMClassifier**

Les modèles **SGDClassifier** et **Logistic regression** ont des performances similaires

Le modèle **RandomForestClassifier** est en sur-apprentissage

**Le Fbeta\_score et le rappel sont maximisés pour les modèles LGBMClassifier et LogisticRegressor**

Le modèle **SGDClassifier** a des performances correctes

Les modèles **XGBoost** et **RandomForestClassifier** ne semblent pas adaptés

Le modèle **DummyClassifier (baseline)** sanctionne fortement les solvables qu'il prédit non-solvable

**LGBMClassifier, SGDClassifier et GaussianNB sont les plus rapides**

RandomForest est un peu plus long en temps de calcul

**Modèle le plus performant :  
LGBMClassifier**

# Optimisation du modèle : LGBM

## GridsearchCV :

Évaluer la meilleure combinaison d'hyper-paramètres

## Objectif :

Augmenter les performance du modèle en optimisant la métrique sélectionné

## Resultat:

Amélioration du Fbeta\_score :

Avant : **0.423611**

Après : **0.431425**

```
# Créer la recherche des hyperparamètres
```

```
params_grid_lgbm = {"n_estimators": [100, 500],  
                    "max_depth": [8, 12],  
                    "learning_rate": [0.01, 0.02],  
                    "num_leaves": [20, 30, 50]  
}
```

```
# Créer le Grid Search
```

```
gscv_lgbm = GridSearchCV(model_lgbm,  
                          params_grid_lgbm,  
                          cv=cv,  
                          scoring=scoring,  
                          refit="fbeta_score",  
                          verbose=True,  
                          n_jobs=-1)
```

# Optimisation et Evaluation : LGBM

## Matrice de confusion

La matrice de confusion mesure les erreurs de prédictions du modèles par rapport à la classification réelle



**VN** : Prédit non-solvable et réellement non-solvable

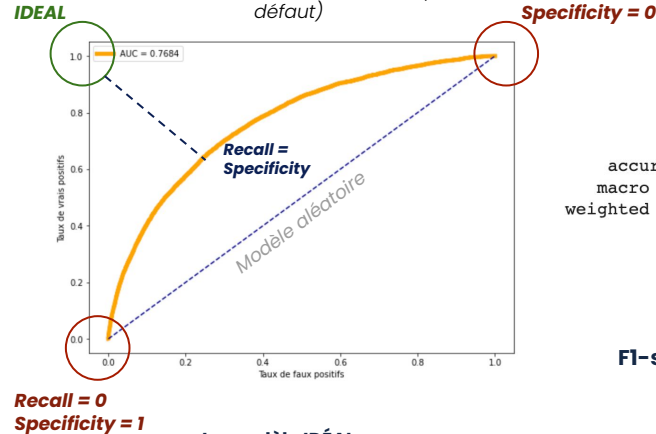
**FN** : Prédit non-solvable MAIS solvable en réalité

**FP** : Prédit solvable MAIS non solvable en réalité

**VP** : Prédit solvable et réellement solvable

## Courbe ROC

La courbe ROC représente le Recall et la Specificity en fonction du seuil de classification (ici, threshold = 0,5 par défaut)



Le modèle IDÉAL a :

un recall = 1

une specificity = 1

## Rapport de classification

Le rapport de classification analyse les métriques par classe

	precision	recall	f1-score	support
0	0.96	0.71	0.82	56535
1	0.17	0.69	0.28	4965
accuracy			0.71	61500
macro avg	0.57	0.70	0.55	61500
weighted avg	0.90	0.71	0.77	61500

$$\text{Recall} = \text{VP} / (\text{VP} + \text{FN})$$

$$\text{Precision} = \text{VP} / (\text{VP} + \text{FP})$$

$$\text{F1-score} = 2 * (\text{Recall} * \text{precision}) / (\text{Recall} + \text{precision})$$

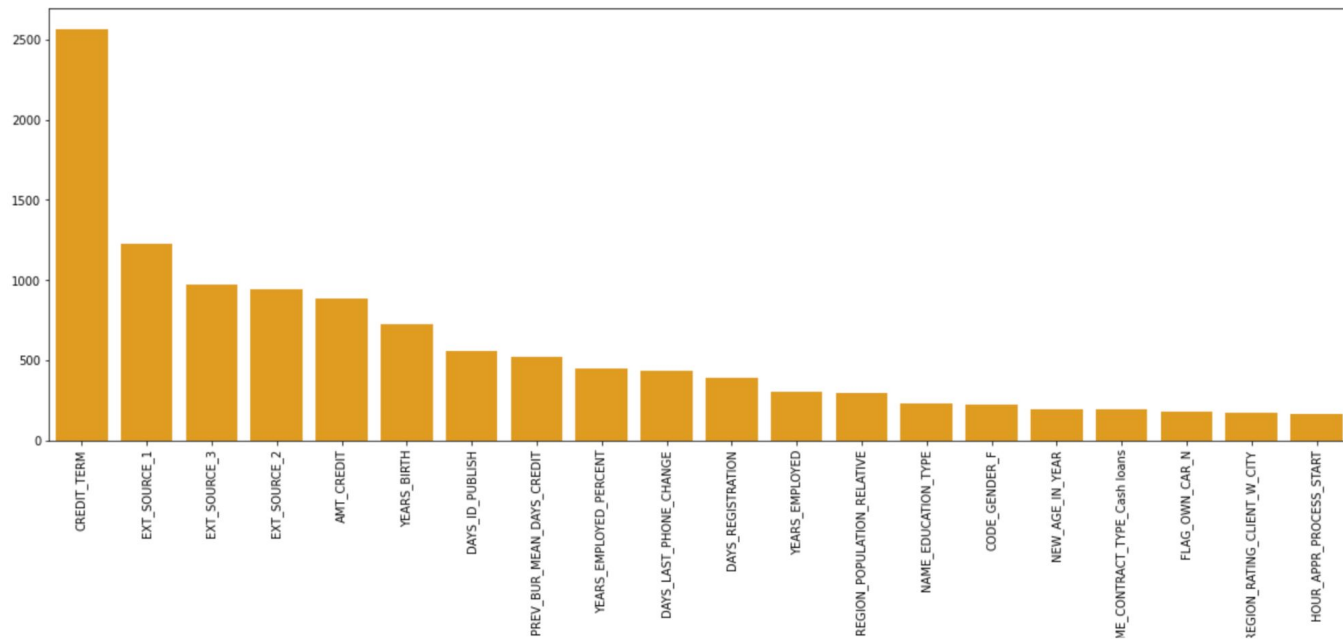
$$\text{Specificity} = \text{VN} / (\text{VN} + \text{FP})$$

$$1 - \text{Specificity} = \text{FP} / (\text{FP} + \text{VN})$$

**Specificity** : capacité du modèle à détecter les clients solvables

Le modèle a tendance à sanctionner les clients solvables

# Features importances



# Tableau des prédictions

Tableau des risques liés aux erreurs de faux négatifs et positifs

	SK_ID_CURR	REEL	PREDICTION	RESULTAT	%_RISQUE_DEFAULT_PAIEMENT	
	0	100001	0	1	REFUSEE	0.57
	1	100005	1	0	ACCORDEE	0.43
	3	100028	1	0	ACCORDEE	0.11
	6	100057	0	1	REFUSEE	0.50
	8	100066	0	1	REFUSEE	0.57
	...	...	...	...	...	...
	48731	456123	0	1	REFUSEE	0.63
	48735	456169	0	1	REFUSEE	0.67
	48737	456189	0	1	REFUSEE	0.58
	48738	456202	0	1	REFUSEE	0.60
	48741	456223	0	1	REFUSEE	0.75

14169 rows x 5 columns

Tableau finale

	SK_ID_CURR	REEL	PREDICTION	RESULTAT	%_RISQUE_DEFAULT_PAIEMENT
0	100001	0	1	A ETUDIER	0.57
1	100005	1	0	A ETUDIER	0.43
2	100013	0	0	ACCORDEE	0.21
3	100028	1	0	A ETUDIER	0.11
4	100038	0	0	ACCORDEE	0.32
...	...	...	...	...	...
48739	456221	0	0	ACCORDEE	0.24
48740	456222	0	0	ACCORDEE	0.19
48741	456223	0	1	A ETUDIER	0.75
48742	456224	0	0	ACCORDEE	0.47
48743	456250	0	0	ACCORDEE	0.31

48744 rows x 5 columns

Je constate que j'ai **14166** lignes qui présentent des risques de faux négatif et de faux positifs.

J'ai pu constater également que les vrais positifs ont un seuil compris entre 20% et 40% de risque de défaut de paiement.

J'ai créer un nouveau tableau avec 3 valeurs pour la variable **RESULTAT** :

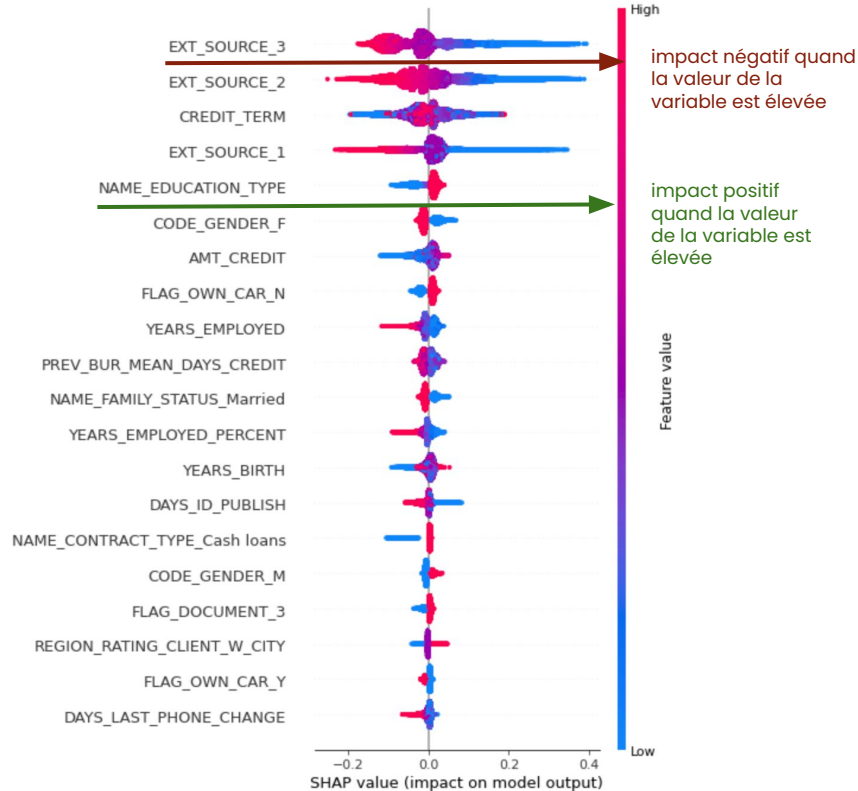
- **ACCORDÉE** (valeur dont la prédiction est égale à 0)
- **REFUSÉE** (valeurs dont la prédiction est égale à 1)
- **À ÉTUDIER** (toutes les valeurs qui présente des risques de faux positifs et de faux négatif)





# Interprétabilité du modèle

# Interpretation Globale



Le diagramme d'importance des variables répertorie **les variables les plus significatives** par ordre décroissant.

Les variables du haut contribuent davantage au modèle que celles du bas et ont donc un pouvoir prédictif élevé.

Le graphique de tracé récapitulatif (**summary\_plot**) est l'un des graphiques les plus importants de la bibliothèque SHAP car il représente la distribution des 20 caractéristiques (variables) les plus importantes de notre modèle, et comment chacune d'elles affecte les résultats du modèle.

Il affiche les informations suivantes :

- **Importance des caractéristiques** : les variables sont classées par ordre décroissant;
- **Impact** : l'emplacement horizontal indique si l'effet de cette valeur est associé à une prédiction supérieure ou inférieure;
- **Valeur d'origine** : la couleur indique si cette variable est élevée (en rouge) ou faible (en bleu) pour cette observation;

**SHAP** calcule l'importance de chaque variable pour chaque donnée et fait la moyenne de l'importance pour obtenir l'importance "globale" de chaque variable.

*Chaque feature peut augmenter ou diminuer la probabilité de défaut de paiement, en fonction du sens de leur influence.*

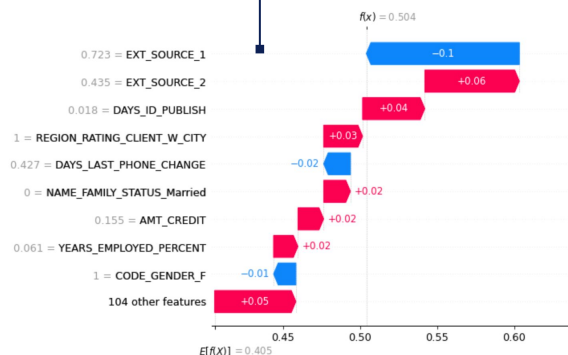
**Ex: si la couleur rouge est à gauche de 0, cela veut dire que plus la valeur de la variable diminue, plus elle va contribuer à prédire un défaut de paiement**

# Interpretation Locale

REFUSER



ACCORDER

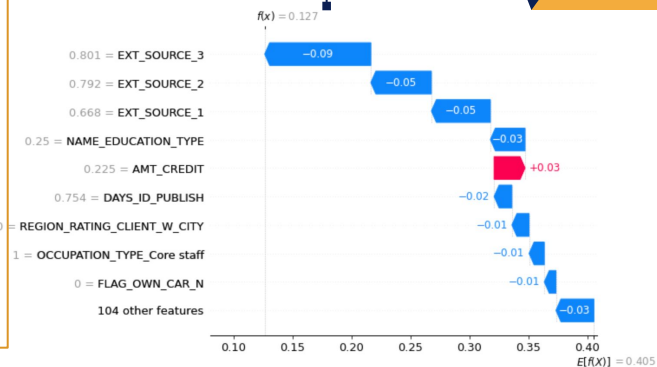


Les graphiques représentent l'impact des variables importantes sur la **prédiction de probabilité de défaut** pour un client donné (=local).

La **valeur de base** (base value) est la moyenne de prédiction de tous les individus.

**Dossier refusé** : les valeurs des variables en rouge contribuent à augmenter la probabilité de défaut.

**Dossier accepté** : les valeurs des variables en bleu contribuent à baisser la probabilité de défaut.





# Conclusions

### Nous avons mis en place :

- Une sélection des métriques
- Un rééquilibrage des données
- L'optimisation des hyper-paramètres

*le résultat du modèle dépend fortement des transformations et des traitements effectués, ainsi que du choix des paramètres de la méthode et du modèle lui-même.*

### les performances peuvent être améliorer :

- **En travaillant avec le métier** pour acquérir une meilleure compréhension de l'entreprise afin que les données puissent être mieux préparées;
- **En approfondissant le mécanisme interne de chaque modèle** pour mieux optimiser les paramètres qui influe le modèle ;
- **En essayant différentes techniques** pour la sélection des fonctionnalités, le rééquilibrage des données (suréchantillonnage/sous-échantillonnage), d'autres outils d'interprétation (par exemple : Lime );

### Pour améliorer la précision de notre modèle :

- **L'ajout de nouvelles fonctionnalités** permettrait une division plus claire entre les clients qui remboursent et ceux qui ne le font pas.

# MERCI !

Avez-vous des questions ?

