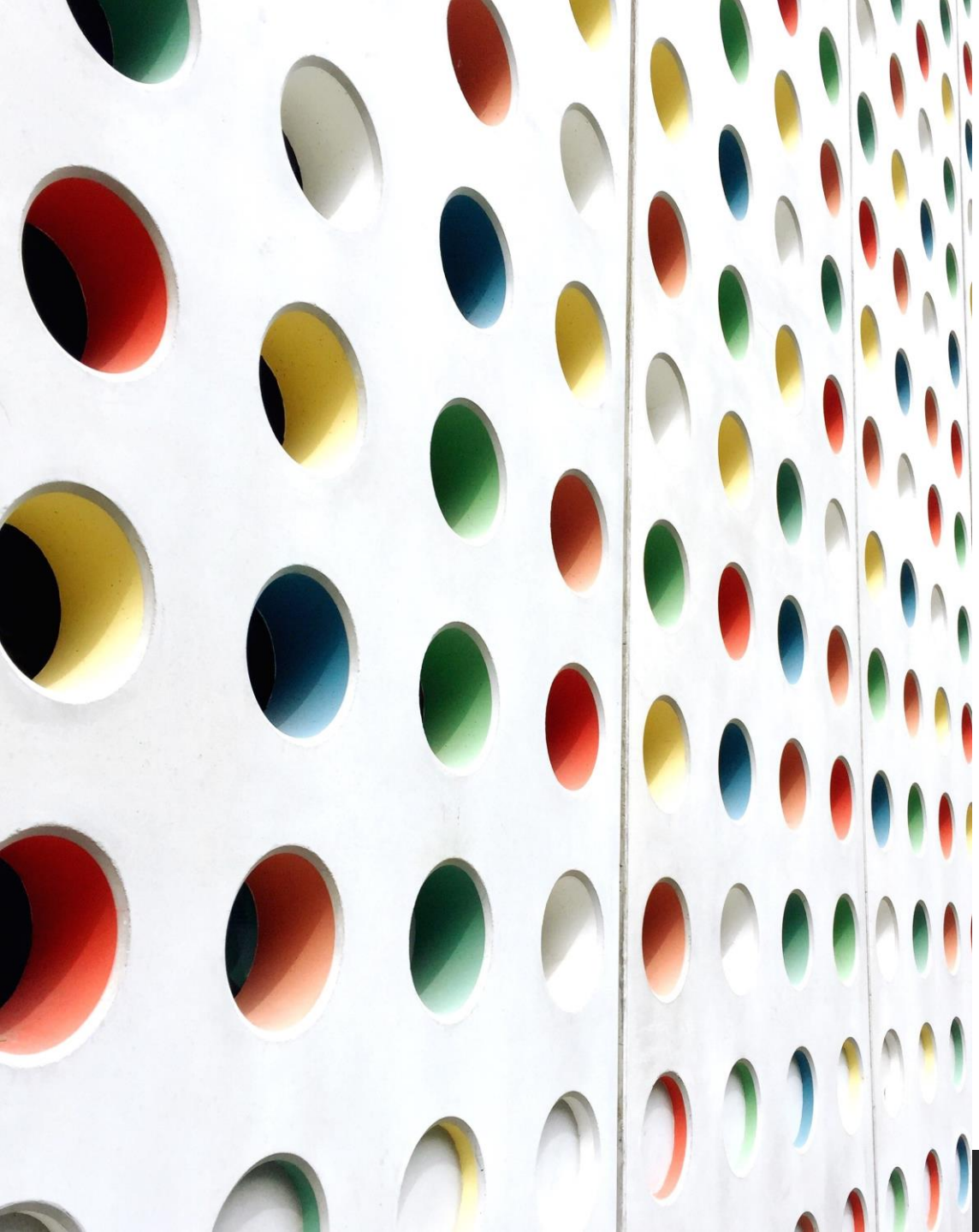


Implémentez un modèle de scoring

PARCOURS DATA SCIENTIST
FINANCÉ PAR PÔLE EMPLOI

MENTOR: GAËTAN GOLLIOT

ÉVALUATEUR: AMINATA DIABY

ÉTUDIANTE: VERONIKA BEREZHNAIA

Sommaire

RAPPEL DE LA PROBLÉMATIQUE ET PRÉSENTATION DU JEU DE DONNÉES (5 MINUTES)

EXPLICATION DE L'APPROCHE DE MODÉLISATION (10 MINUTES)

PRÉSENTATION DU DASHBOARD (5 MINUTES)

Problématique

Mon entreprise propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt.

Elle souhaite développer un algorithme pour calculer la probabilité qu'un client rembourse son crédit, puis classifie la demande en crédit accordé ou refusé.

Les clients et l'entreprise demandent de demandeurs de transparence vis-à-vis des décisions d'octroi de crédit.

Ma mission

Construire un modèle de scoring et un dashboard interactif.

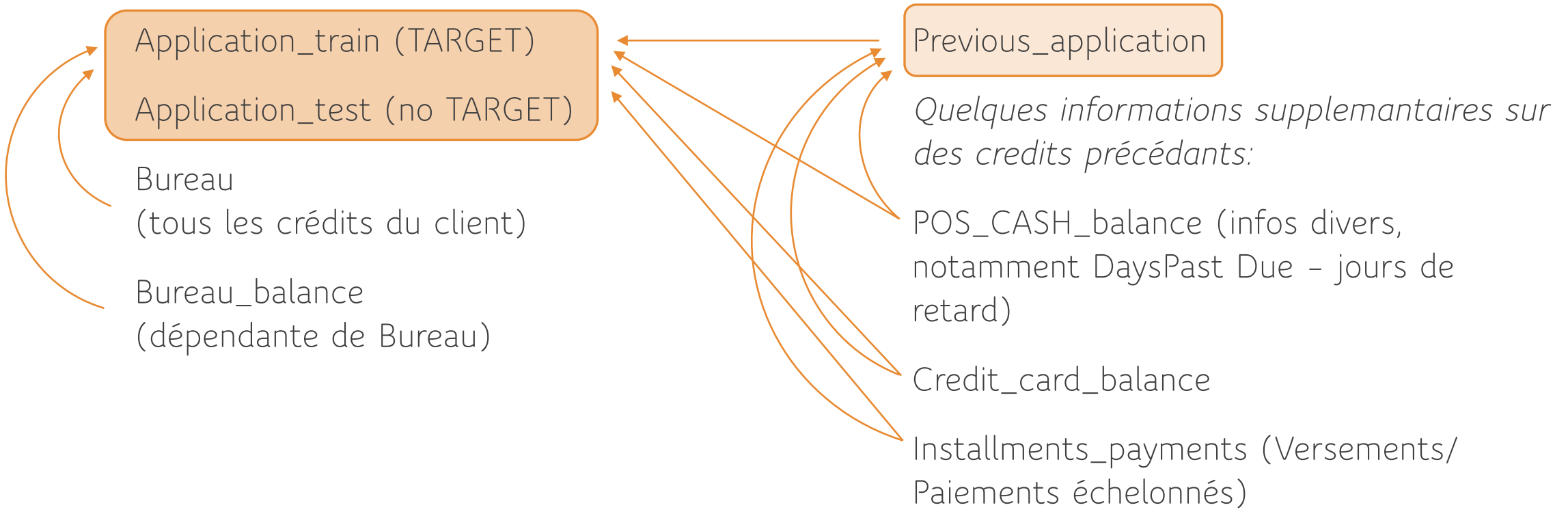
Spécifications du dashboard:

Visualiser le score et l'interprétation de ce score

Visualiser des informations descriptives relatives à un client

Comparer les informations descriptives relatives à un client à l'ensemble des clients

Jeu de données



L’explication de l’approche de modélisation

L’analyse des données exploratoire:
la sélection de fonctionnalités (features),
l’imputation des valeurs manquantes

- Application_train
- Bureau et Bureau_balance

L’entraînement du modèle, l’algorithme d'optimisation:

- La gestion des classes déséquilibrés
- Le choix du modèle
- Réglage des hyperparamètres
- Métrique d’évaluation
- Fonction coût métier

L’interprétabilité globale et locale du modèle

L’API et le tableau de bord

Des limites et des améliorations possibles

L’analyse des données exploratoire

Type Object

Type Flags

Type Integer

Type Float

NaN	96391
Laborers	55186
Sales staff	32102
Core staff	27570
Managers	21371
Drivers	18603
High skill tech staff	11380
Accountants	9813
Medicine staff	8537
Security staff	6721
Cooking staff	5946
Cleaning staff	4653
Private service staff	2652
Low-skill Laborers	2093
Waiters/barmen staff	1348
Secretaries	1305
Realty agents	751
HR staff	563
IT staff	526

TARGET	0	1	proportion
missing_value	93.486944	6.513056	31.345545
Laborers	89.421230	10.578770	17.946025
Sales staff	90.368201	9.631799	10.439301
Core staff	93.696046	6.303954	8.965533
Managers	93.785972	6.214028	6.949670
Drivers	88.673870	11.326130	6.049540
High skill tech staff	93.840070	6.159930	3.700681
Accountants	95.169673	4.830327	3.191105
Medicine staff	93.299754	6.700246	2.776161
Security staff	89.257551	10.742449	2.185613
Cooking staff	89.556004	10.443996	1.933589
Cleaning staff	90.393295	9.606705	1.513117
Private service staff	93.401207	6.598793	0.862408
Low-skill Laborers	82.847587	17.152413	0.680626
Waiters/barmen staff	88.724036	11.275964	0.438358
Secretaries	92.950192	7.049808	0.424375
Realty agents	92.143808	7.856192	0.244219
HR staff	93.605684	6.394316	0.183083
IT staff	93.536122	6.463878	0.171051

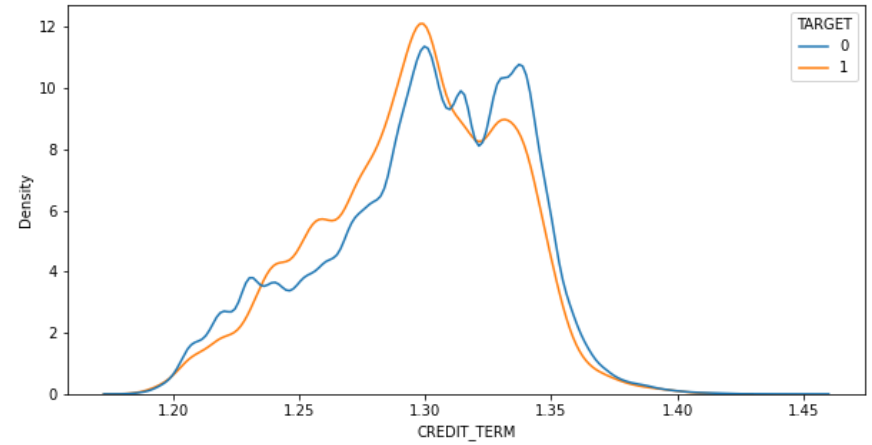
L'analyse des données exploratoire

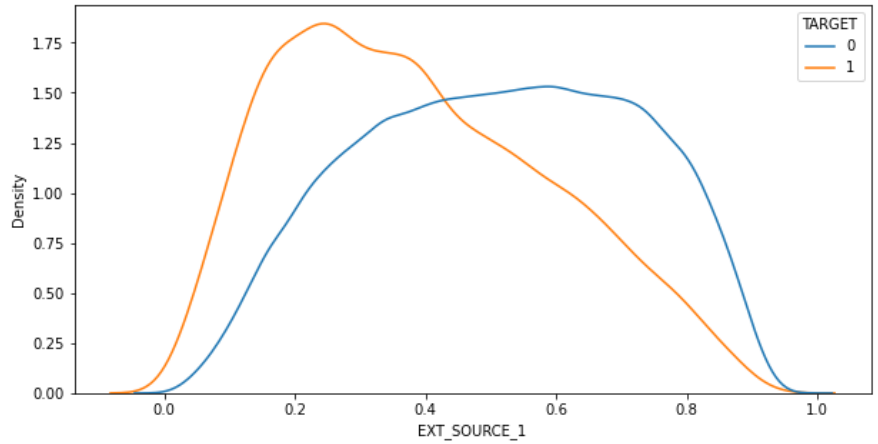
Type Object

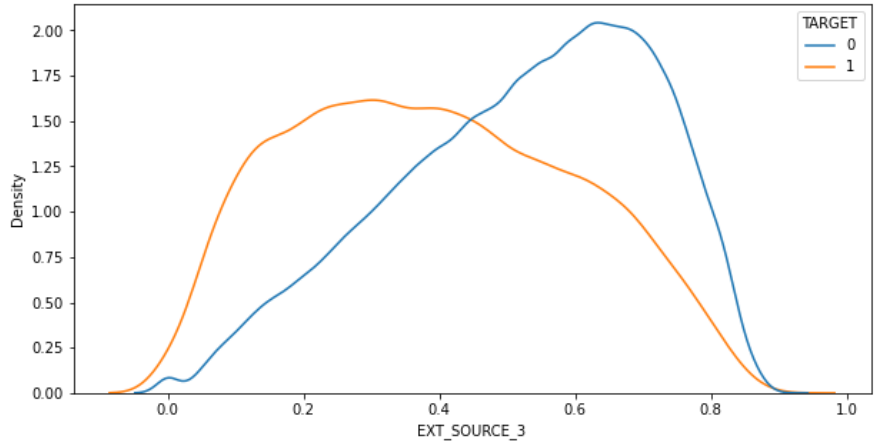
Type Flags

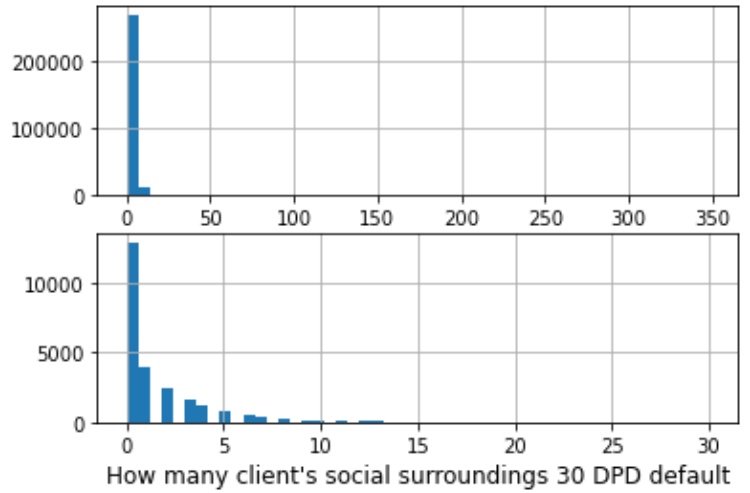
Type Integer

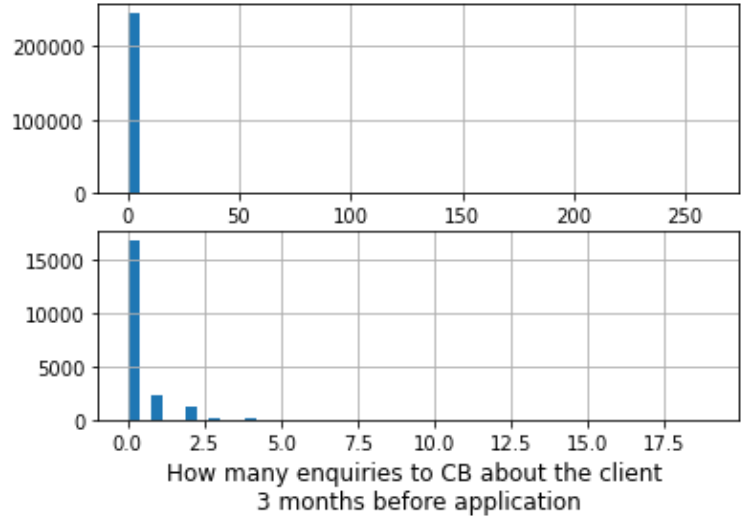
Type Float











Dependent tables

Bureau_balance:

Analyser (pas de valeurs manquantes)

Aggréger

Joindre à Bureau

Imputer des valeurs manquantes venants du bureau_balance:

dummies avec 1 sur inconnu et 0 ailleurs

Bureau:

Analyser

Imputer des valeurs manquantes

Aggréger

Joindre à application_train

Imputer des valeurs manquantes venants du Bureau

Préparation finale

Tailler le jeu de données pour réduire le temps d'exécution de notebook

Comment:

- Lancer le notebook avec l'ensemble entier des variables
- Tenter d'éliminer toutes les valeurs sauf les plus pertinents
- Voir à quel point tombe la performance du modèle (4%)

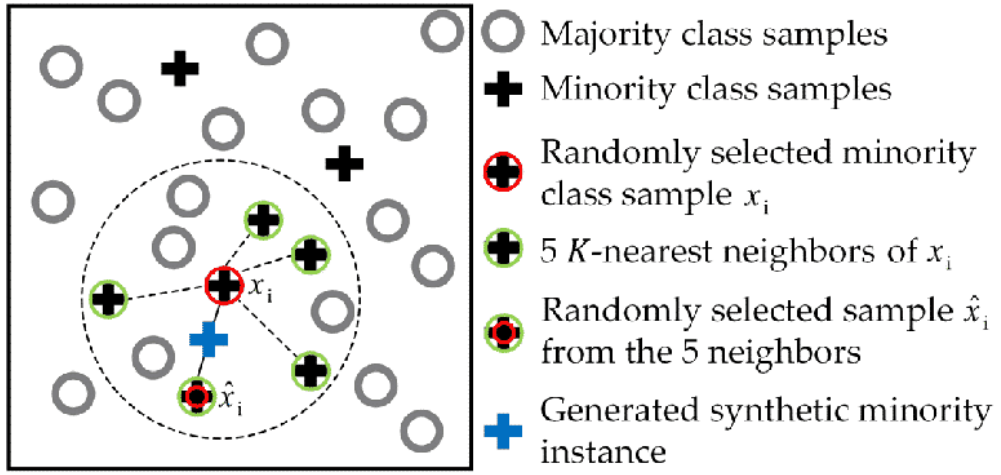
Gestion des classes deséquilibrées

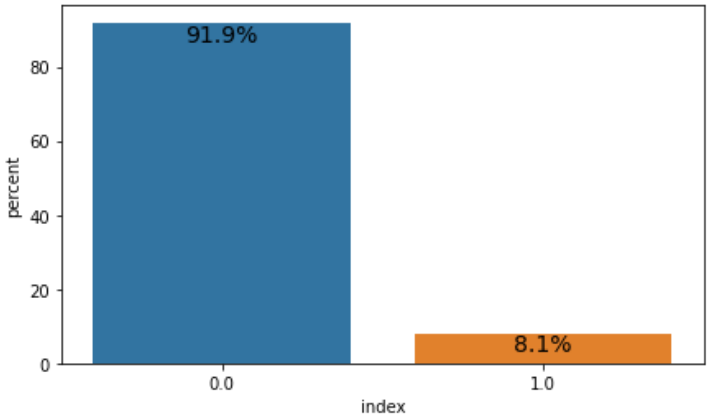
1. Class_weight = 'balanced'

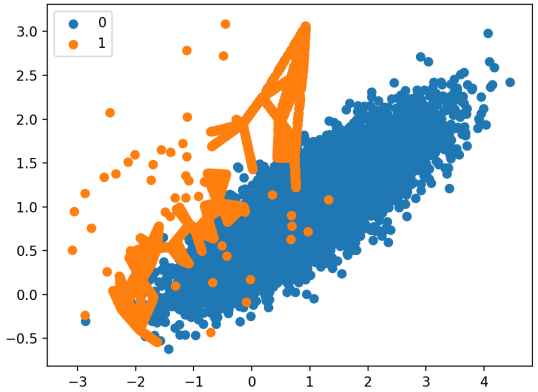
ou scale_pos_weight = class1 / classe0 pour XGB

2. SMOTE, une technique de oversampling: encombre la mémoire

3. Il faut essayer avec des techniques de undersampling (Tomek Links, ENN)







Le choix du modèle et le réglage des hyperparamètres

Algorithmes natifs SKLearn :

Logistic Regression

Passive Aggressive Classifier

Random Forest Classifier

Les tous sont testés
avec class_weight = 'balanced'
et avec SMOTE

Algorithmes provenant de XGBoost et enveloppés dans le shell SKLearn:

XGBClassifier(ne possède pas de paramètre class_weight)

LightGBMClassifier (testé uniquement avec class_weight = 'balanced' mais pas avec SMOTE – cela serait trop lent)

GridSearchCV (exhaustive GridSearch). Tester d'autres strategies?

La sélection du meilleur modèle et la fonction coût métier

Accuracy estime Dummy Classifier à 92% - ne convient pas pour des classes désequilibrées

Comparer des modèles par ROC_AUC (FN vs FP)

Le meilleur modèle est Logistic Regression avec class_weight 'balanced'

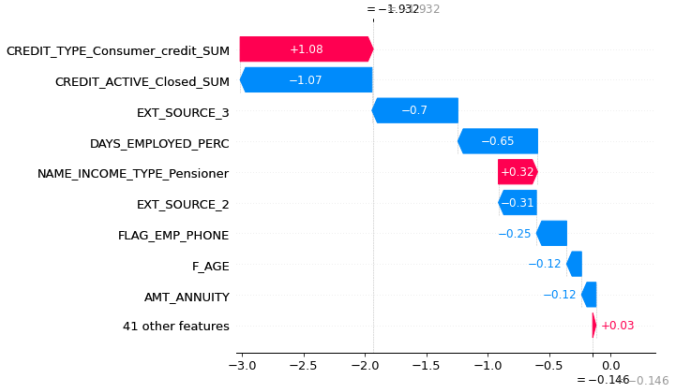
La fonction coût métier prend en compte le coût d'un faut positif et d'un faut négaitf, avec ses coefficients

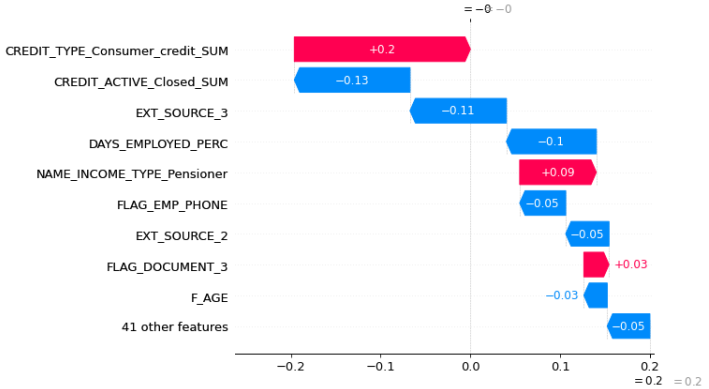
	Model	Source	SMOTE	Perso	AUC	Recall	Precision
6	LogReg	cv_train	False	-0.557	0.741	0.677	0.156
3	PassAggr	cv_train	False	-0.557	0.740	0.674	0.157

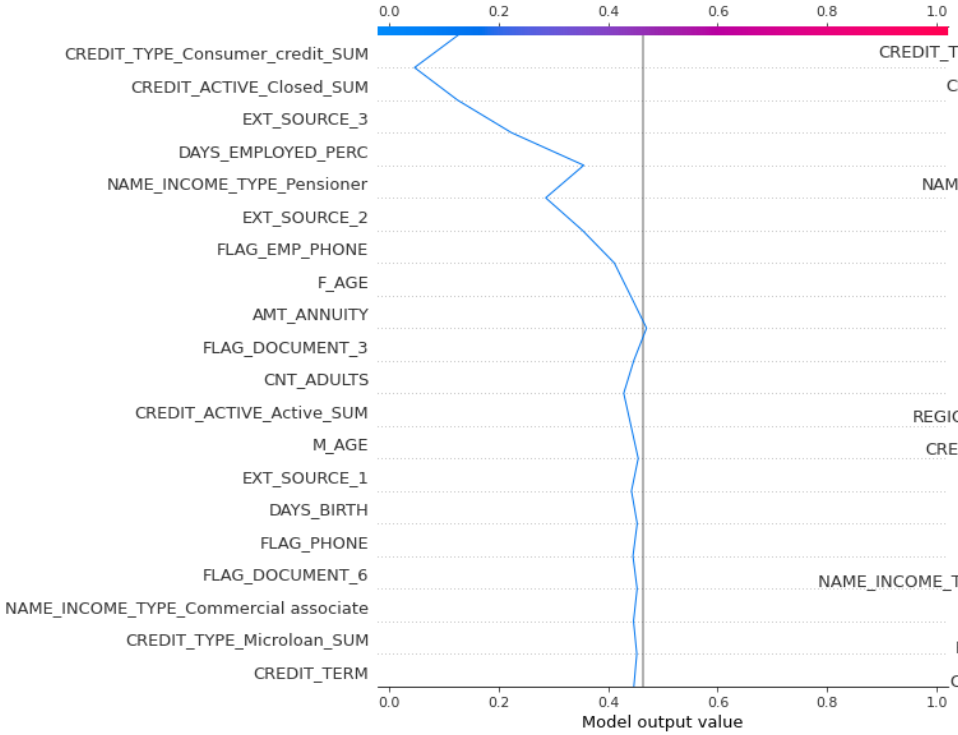
	Model	Source	SMOTE	Perso	AUC	Recall	Precision
7	LogReg	cv_test	False	-0.558	0.740	0.675	0.156
4	PassAggr	cv_test	False	-0.558	0.739	0.673	0.156

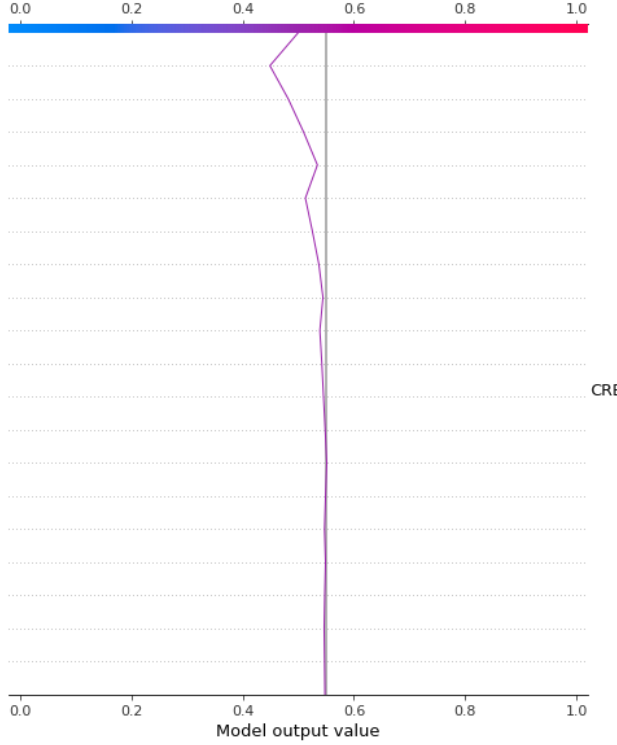
	Model	Source	SMOTE	Perso	AUC	Recall	Precision
8	LogReg	external_test	False	-0.555	0.679	0.678	0.157
5	PassAggr	external_test	False	-0.556	0.678	0.676	0.156

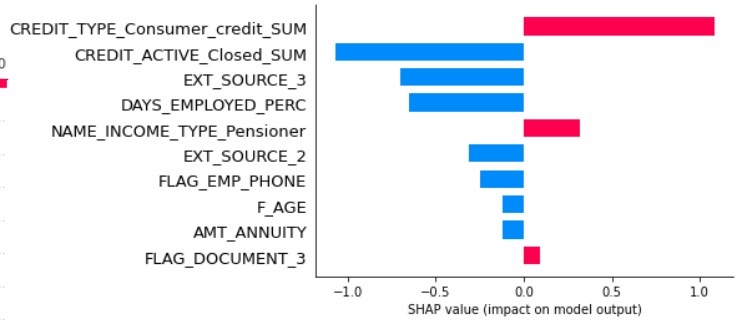
L'interprétabilité locale du modèle

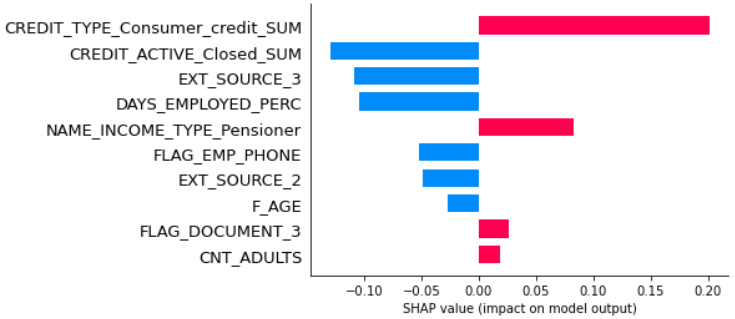




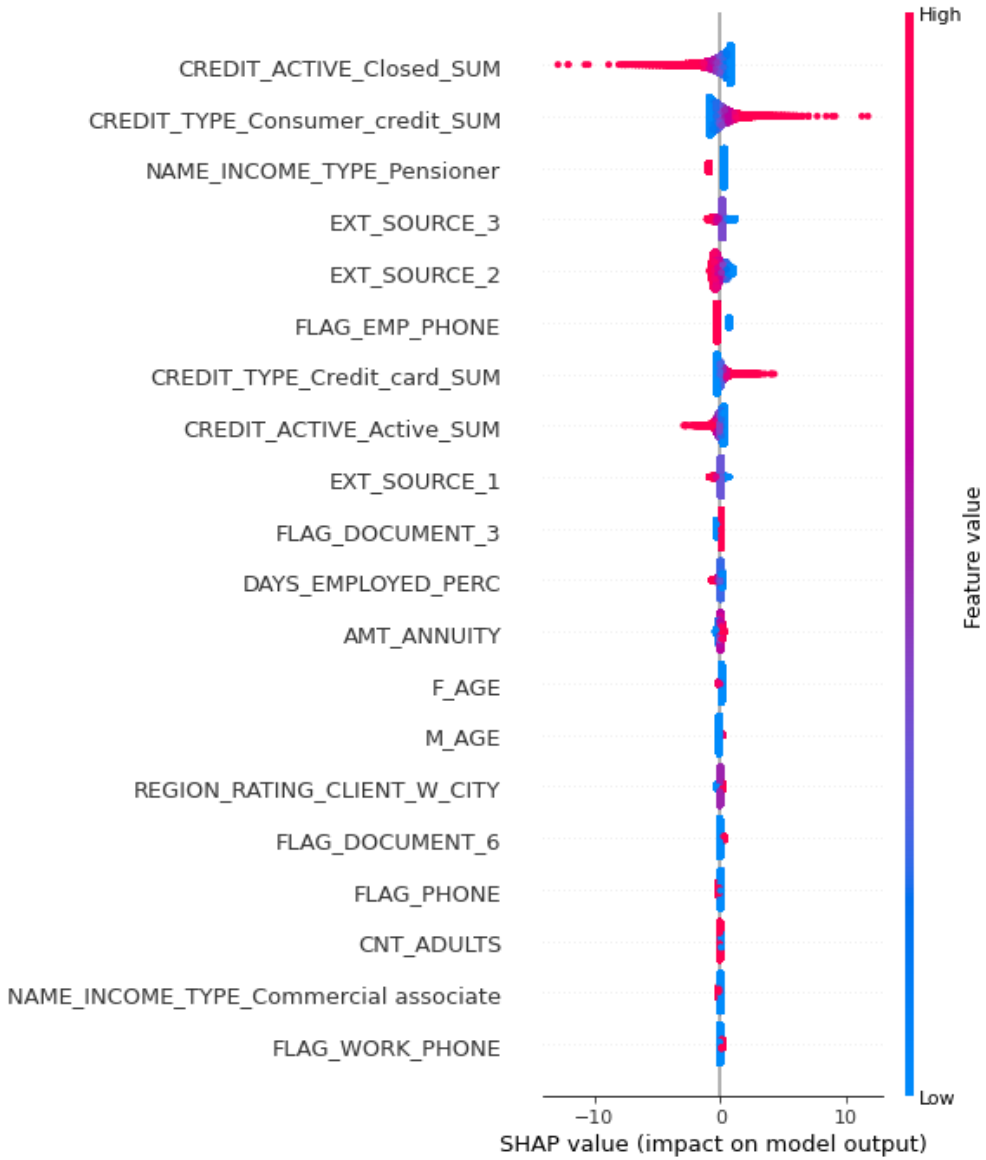








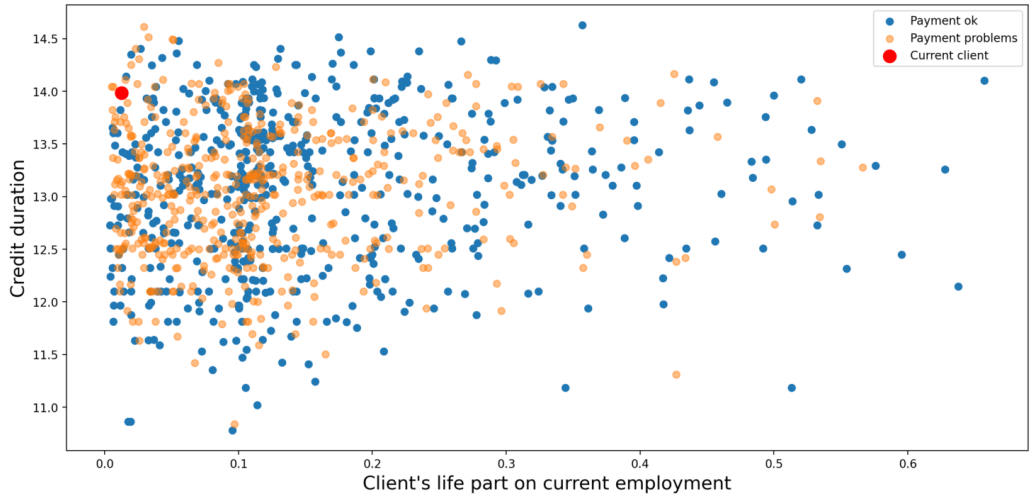
L'interprétabilité globale du modèle (summary plot)

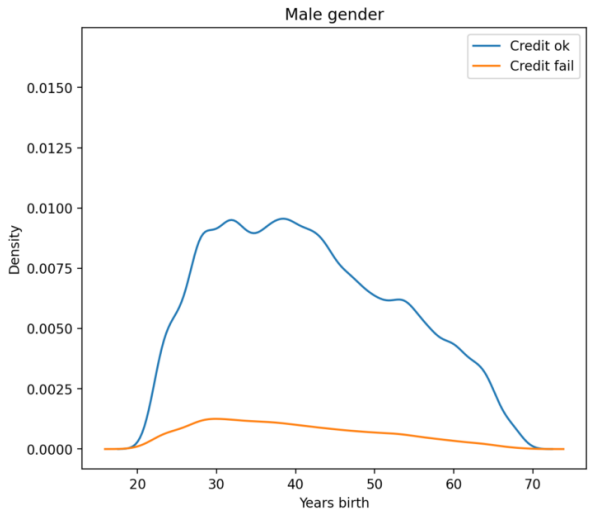


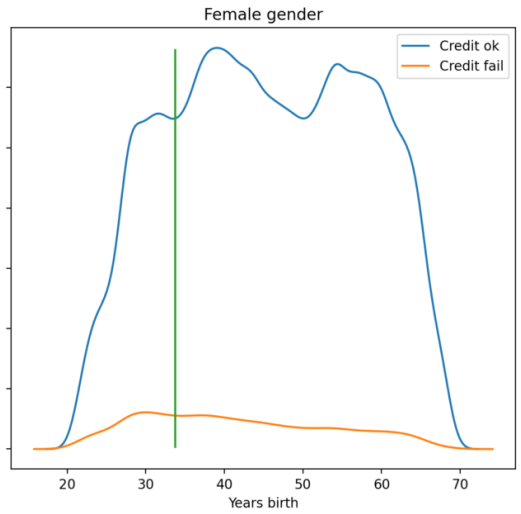
L'API et le dashboard

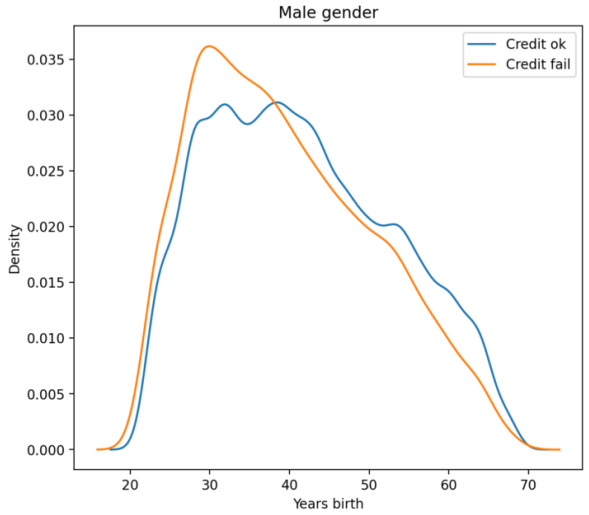
L'API était developpé avec MLFlow

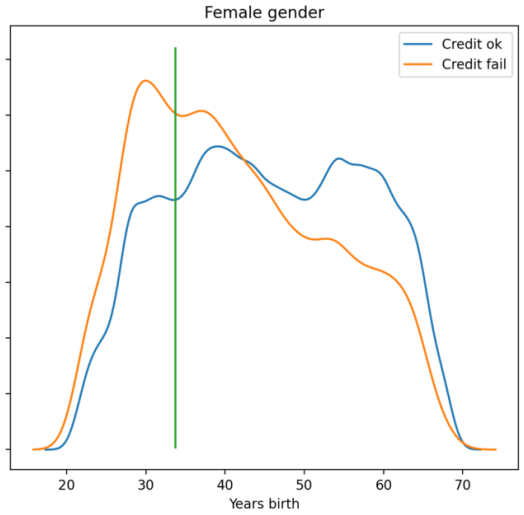
Il est possible de saisir les données manuellement ou automatiquement











Conclusion

Des améliorations:

Choisir le seuil de probabilité optimal pour la séparation des classes

Insérer la recherche de ce seuil dans l'entrainement du pipeline, comme l'un des hyperparamètres

Faire de sorte que la sortie du modèle soit la probabilité et non pas une classe

Rajouter des techniques de undersampling (Tomek Links, ENN)

Ajouter la fonction «column to fail proportion» dans le pipeline (ColumnTransformer)

Trouver des méthodes pour économiser le temps d'entraînement des modèles

Montrer un client sur le fond des clients similaires (et d'où prendre ces clients similaires)

Le modèle de prédiction de score de défaut de crédit a été construit

Des meilleurs parmètres sélectionnés: façon de gérer des classes déséwuilibrées, le modèle et ses hyperparamètres.

L'API de prédiction était déployé (sur mon ordinateur)

Le dashbord interactif a été déployé sur le serveur Héroku

Permettre de visualiser le score et l'interprétation de ce score pour chaque client de façon intelligible pour une personne non experte en data science.

Permettre de visualiser des informations descriptives relatives à un client (via un système de filtre).

Permettre de comparer les informations descriptives relatives à un client à l'ensemble des clients ou à un groupe de clients similaires.