Projet 7

Implémentez un modèle de scoring

18 Mars 2022

MBIADOU SALEU G. Raïssa

PLAN

01

CONTEXTE

CONTEXTE

🎯 AMÉLIORER L'EXPÉRIENCE DE PRÊT BANCAIRE

- « Prêt à dépenser » : société de crédits à la consommation
- Développer un modèle de scoring pour prédire la probabilité de défaut de paiement d'un client
- Développer un dashboard interactif pour assurer une transparence sur les décisions d'octroi de crédit

LES ENJEUX

Éviter des pertes

Déterminer si les clients potentiels sont capables de rembourser leur crédit pour éviter de perdre de l'argent en prêtant à des mauvais clients

Faire du profit

Eviter de perdre de bons prospects en refusant d'accorder des prêts à des clients qui sont pourtant en mesure de rembourser leur crédit

Transparence

Être capable d'expliquer aux clients de manière claire et transparente la décision d'accord ou de refus d'un crédit

02

PRÉSENTATION DES DONNÉES

LES DONNÉES

	Nom du fichier	Nb de lignes	Nb de colonnes	%NaN	%Duplicate
1	Application Train	307511	122	24.40	0.0
2	Application Test	48744	121	23.81	0.0
3	Bureau	1716428	17	13.50	0.0
4	Bureau Balance	27299925	3	0.00	0.0
5	Credit Card Balance	3840312	23	6.65	0.0
6	Installments Payments	13605401	8	0.01	0.0
7	POS CASH balance	10001358	8	0.07	0.0
8	Previous Application	1670214	37	17.98	0.0

Base de données bancaires anonymisées

- 8 fichiers (dont 2 principaux)
- Plus de 300000 clients
- 5 types d'informations:
  - Informations clients
  - Historique de prêts
  - Données cartes de crédit
  - Données liquidités
  - Autres informations de traitement

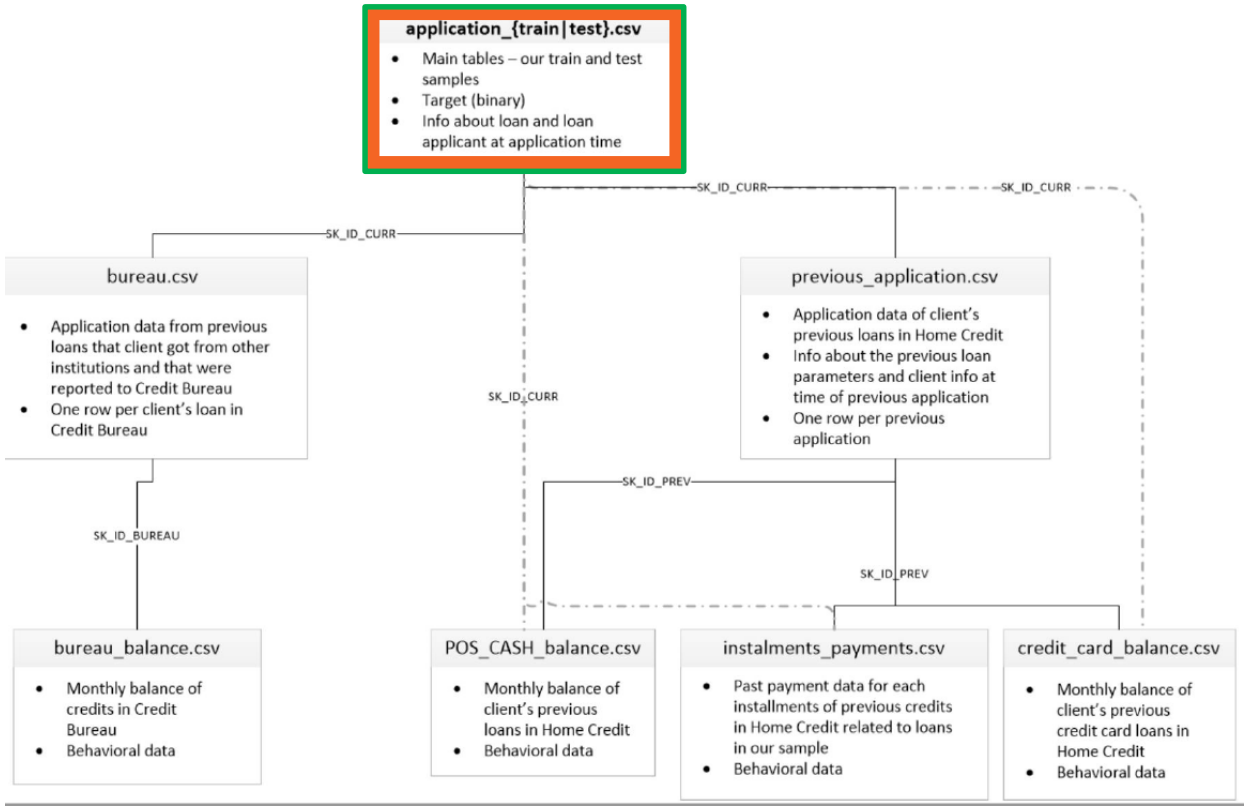
TARGET

Statut de remboursement du prêt

NON-défaillants

Défaillants

RELATIONS ENTRE LES FICHIERS



03
ANALYSE EXPLORATOIRE DES DONNÉES

Environnement de développement

Anaconda

Installation d'Anaconda: plateforme de disbution python la plus populaire

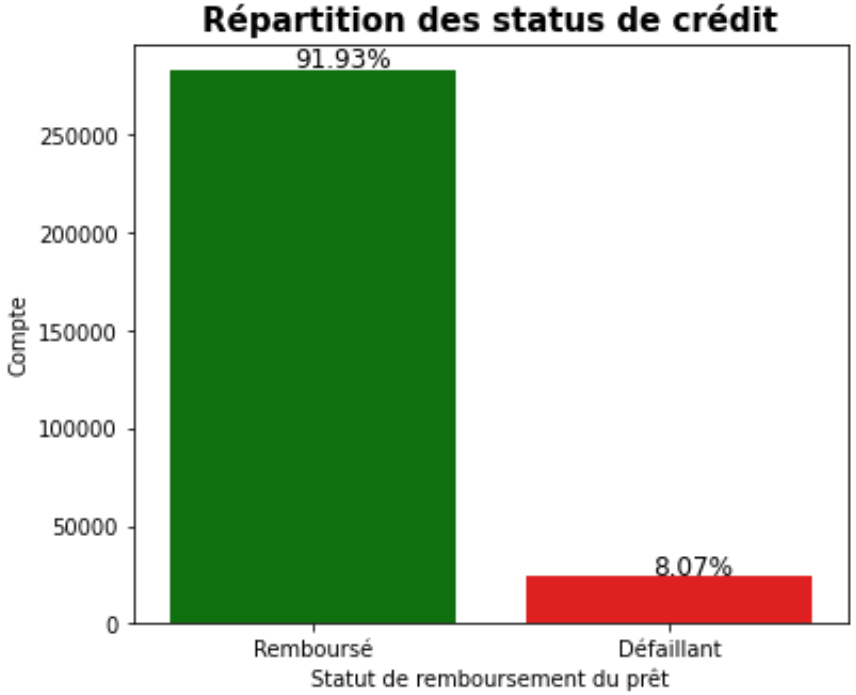
environement virtuel

Mise en place d'un environnement virtuel dédié au projet

Installation des paquets

Installation des paquets nécessaires (numpy, pandas,matplotlib, seaborn, sklearn, scipy, shap, flask, steamlit, plotly, etc...) avec la commande **pip install**

RÉPARTITION DE TARGET

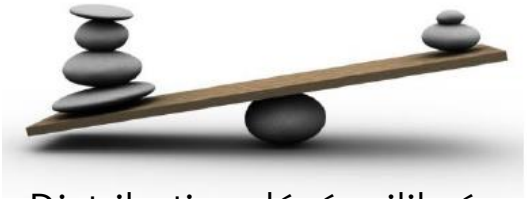


91.9 % **NON-défaillants**

La plupart des clients remboursent leur crédit

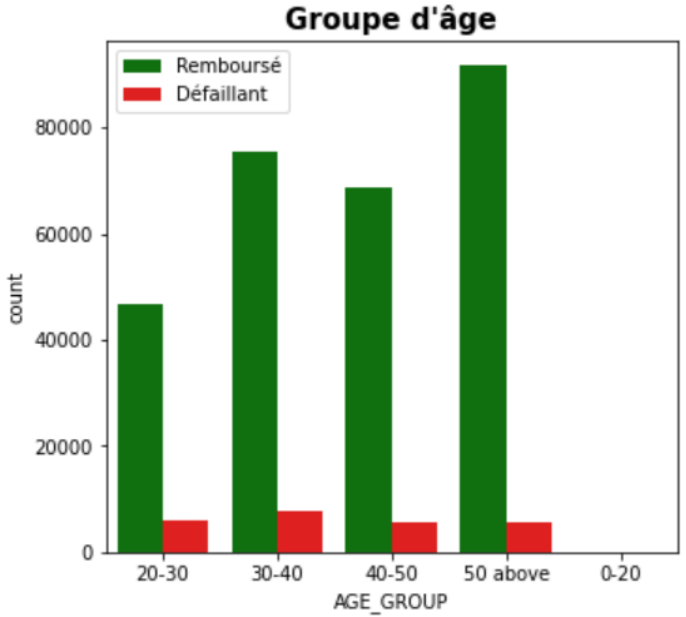
8.1 % **Défaillants**

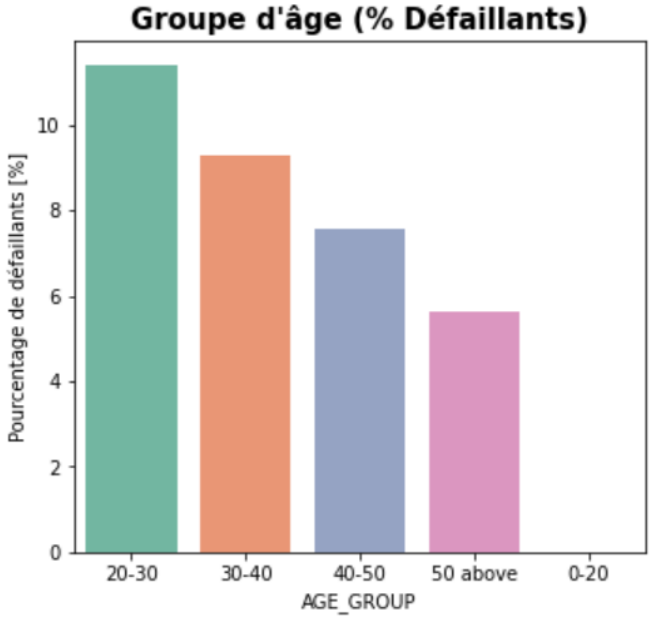
Seulement 8% de clients sont défaillants



Distribution déséquilibrée

TARGET EN FONCTION DE L'ÂGE

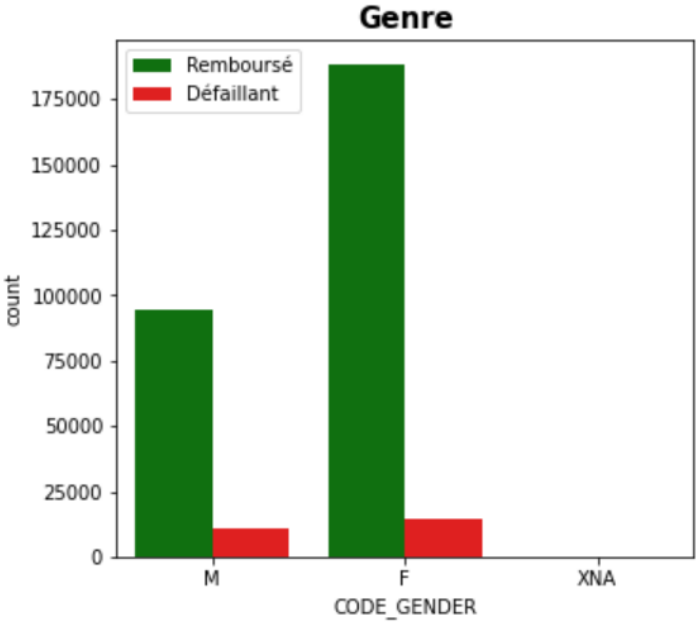


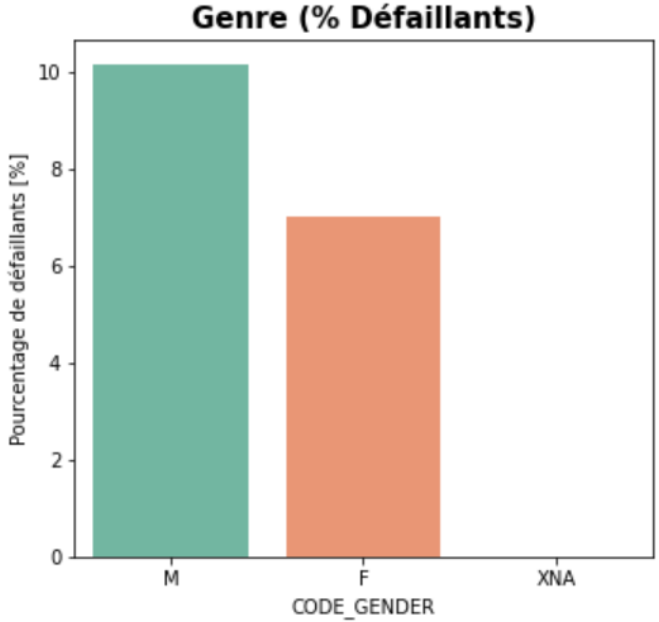


Les personnes de la tranche d'âge 20-40 ans ont une probabilité plus élevée de défaillance.

Les personnes âgées de plus de 50 ans ont une faible probabilité de défaillance.

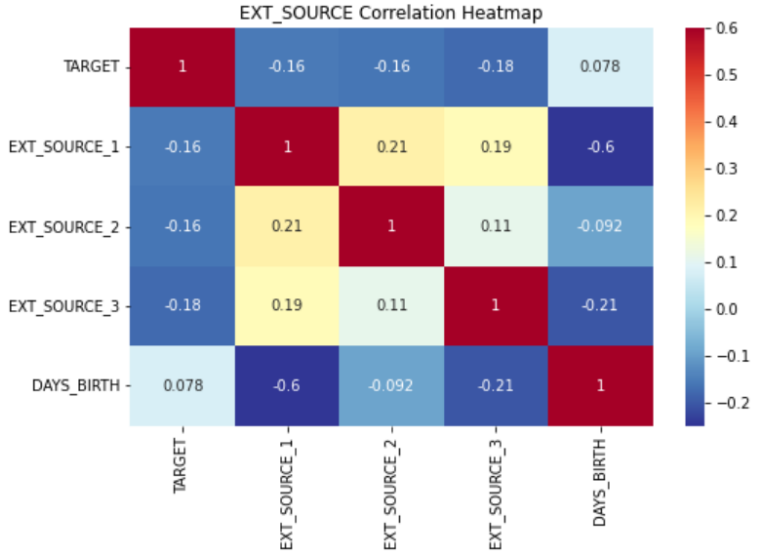
TARGET EN FONCTION DU GENRE

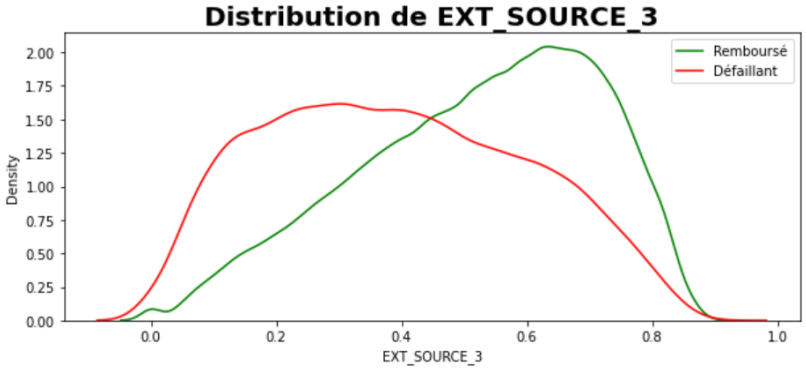




Le nombre de clients féminins est presque le double du nombre de clients masculins.

Les hommes ont plus de chances de ne pas rembourser leurs prêts (~ 10 %), par rapport aux femmes (~ 7 %)





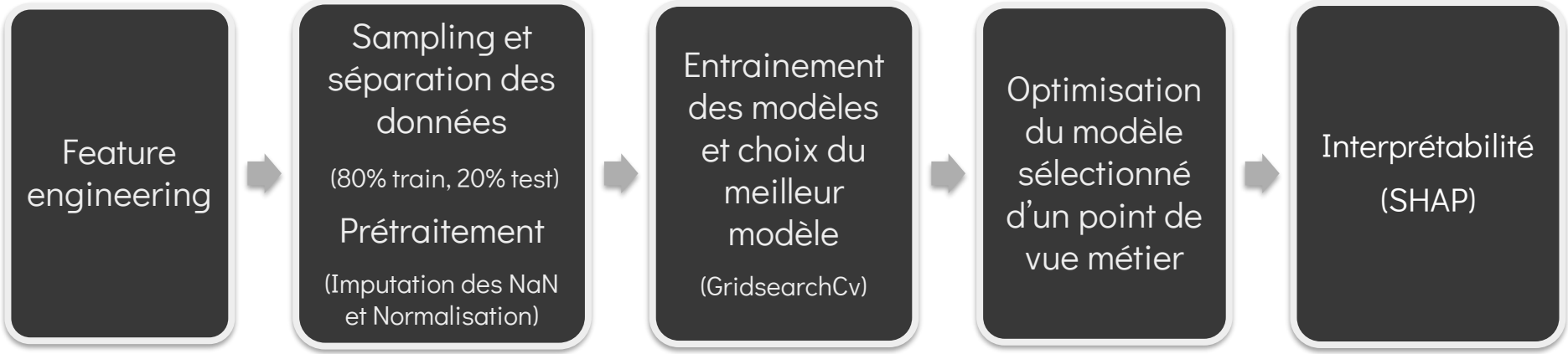
Corrélations

- Les trois variables EXT_SOURCE ont des corrélations négatives avec la variable cible, ce qui indique que plus la valeur de la EXT_SOURCE augmente, plus le client est susceptible de rembourser le prêt.
- Selon la documentation, ces variables représentent un "score normalisé provenant d'une source de données externe"
- EXT_SOURCE_3 présente la corrélation la plus forte avec le remboursement du prêt.

04

Modélisation

DÉMARCHE


Feature engineering
Sampling et séparation des données
(80% train, 20% test)
Prétraitement
(Imputation des NaN et Normalisation)
Entrainement des modèles et choix du meilleur modèle
(GridsearchCv)
Optimisation du modèle sélectionné d'un point de vue métier
Interprétabilité
(SHAP)

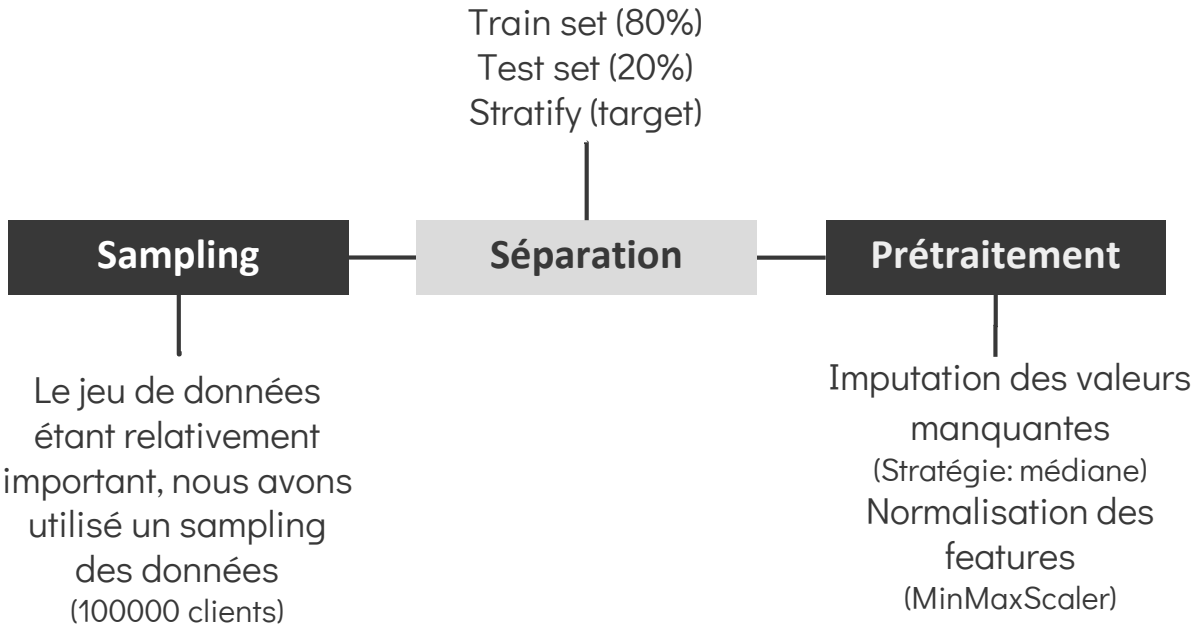
Feature engineering

Cette partie a été entièrement inspirée du notebook kaggle suivant: https://www.kaggle.com/jsaguiar/lightgbm-with-simple-features/

1- Création des features représentant des taux pour certaines variables importantes comme par exemple «Annuity», «Income»
2- Dans «Bureau Data» : création de features spécifiques pour les crédits actifs et les crédits fermés
3- Dans «Previous Applications» : création des features spécifiques pour les candidatures approuvées et refusées
4- Modularité : une fonction de feature engineering pour chaque fichier (sauf bureau_balance et application_test)
5- One-hot encoding pour les variables catégorielles
6- La plupart des fonctionnalités sont créées en appliquant les fonctions min, max, mean, sum et var à des tables groupées
7- Toutes les tables sont jointes à l'aide de la clé SK_ID_CURR (sauf bureau_balance)

796 features

SAMPLING, SÉPARATION, PRÉTRAITEMENT



Modélisation

Algorithmes testés

1- DummyClassifier (baseline)
2- Régression Logistique
3- RandomForestClassifier
4- Light Gradient Boosting Machine

Gestion du déséquilibre

Undersampling
Oversampling (SMOTE)
Weight_balanced (Balanced)

Entrainement

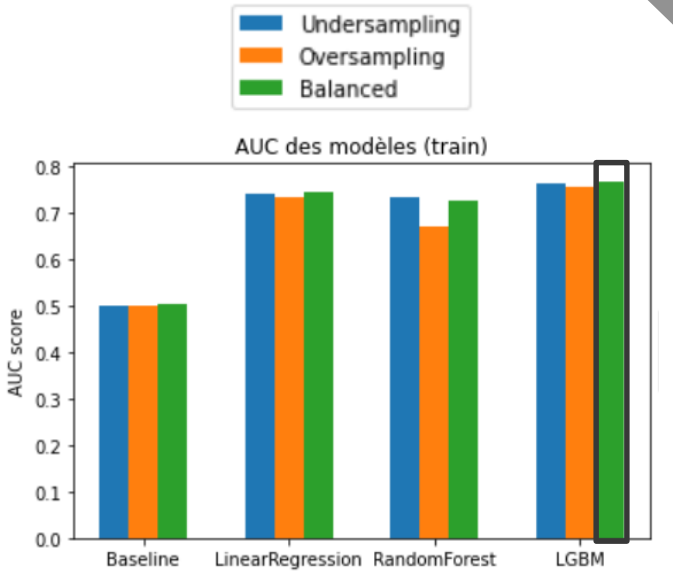
Utilisation d'une validation croisée (5 fold) avec recherche des hyperparamètres optimaux via GridSearch

Métrique d'évaluation

Evaluation des modèles à l'aide de l'AUC (Area Under the ROC Curve)
Plus l'AUC est élevée, plus le modèle est capable de prédire les 0 comme 0 et les 1 comme 1

RÉSULTATS

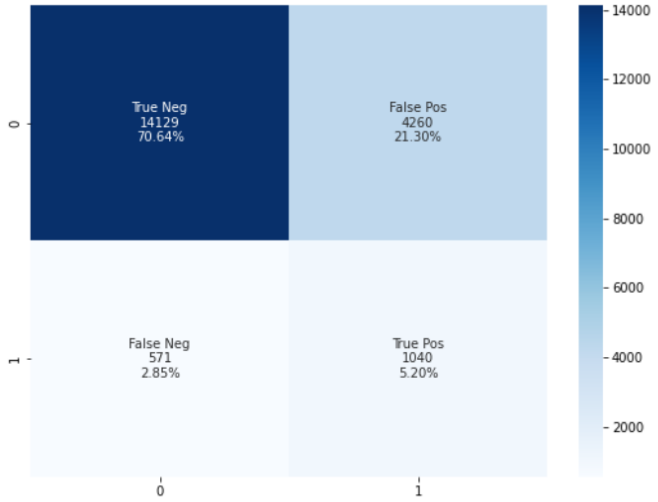
	Algorithm	Balancing_method	AUC	AUC_test	Time
0	Baseline	Undersampling	0.500	0.500	0.015622
1	Baseline	Oversampling	0.500	0.500	0.015620
2	Baseline	Balanced	0.504	0.501	0.000000
3	LogisticRegression	Undersampling	0.741	0.760	67.179755
4	LogisticRegression	Oversampling	0.733	0.754	1309.713699
5	LogisticRegression	Balanced	0.745	0.763	224.687680
6	RandomForest	Undersampling	0.733	0.749	108.216721
7	RandomForest	Oversampling	0.670	0.679	1600.601269
8	RandomForest	Balanced	0.726	0.736	488.666188
9	LGBM	Undersampling	0.762	0.775	122.886276
10	LGBM	Oversampling	0.756	0.773	1618.627453
11	LGBM	Balanced	0.767	0.780	533.581348

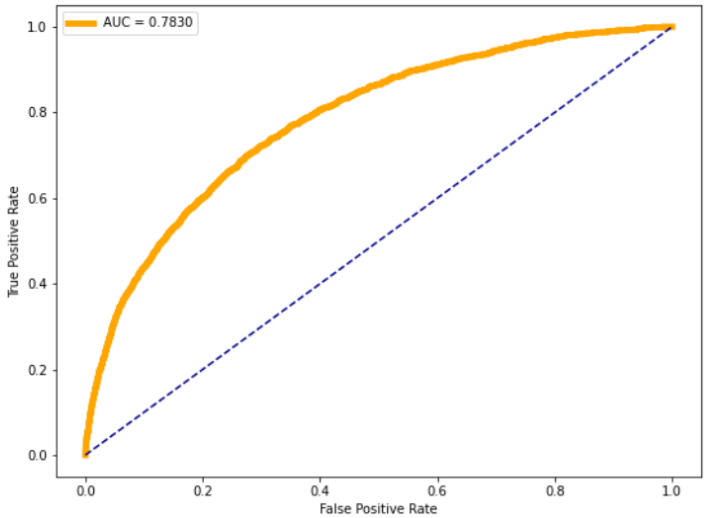


LGBM associé à la stratégie de rééquilibrage consistant à indiquer "balanced" comme valeur pour "class_weight" donne les meilleurs résultats

RÉSULTATS (JEU DE TEST)

LGBM (Balancing Method: Balanced)

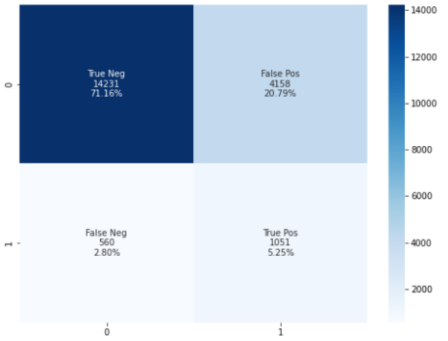


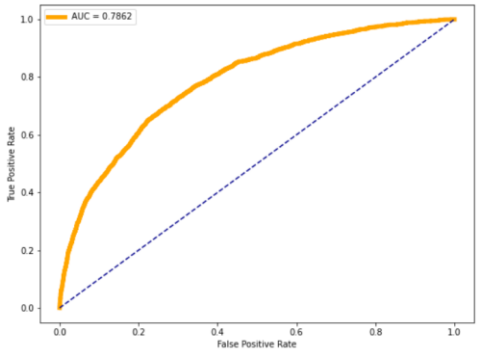


FP : bon client considéré comme mauvais = crédit non accordé à tort, donc manque à gagner de la marge pour la banque

FN : mauvais client à qui on accorde un prêt, donc perte sur le capital non remboursé

HyperOpt optimized LGBM



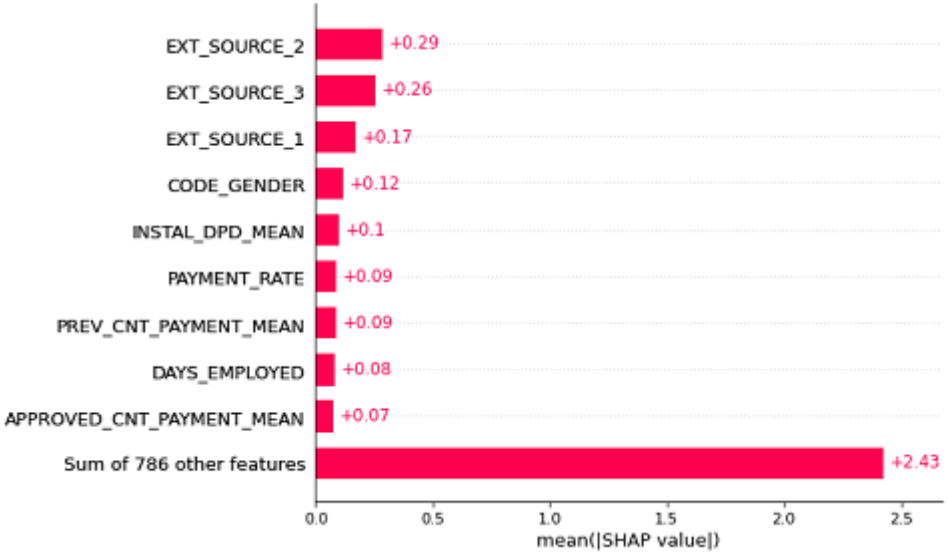


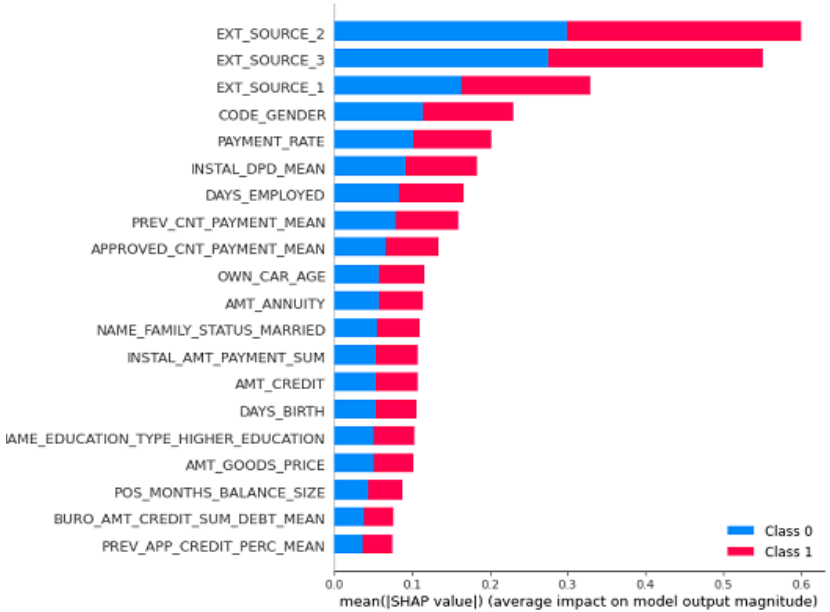
On a un meilleur score AUC et une diminution du nombre de FN sur le jeu de test.

OPTIMISATION MÉTIER

- La problématique « métier » est de prendre en compte qu'un faux positifs n'a pas le même coût qu'un faux négatif
- Un faux négatif est en effet 10 fois plus coûteux qu'un faux positif
- Nous avons défini une fonction métier adaptée qui permet d'attribuer plus de poids à la minimisation des FN
- Nouvelle recherche des hyper-paramètres via HyperOpt se basant sur la fonction métier proposée

EXPLICABILITÉ (FEATURE IMPORTANCE)





Les principales features qui contribuent à l'élaboration du modèle

05

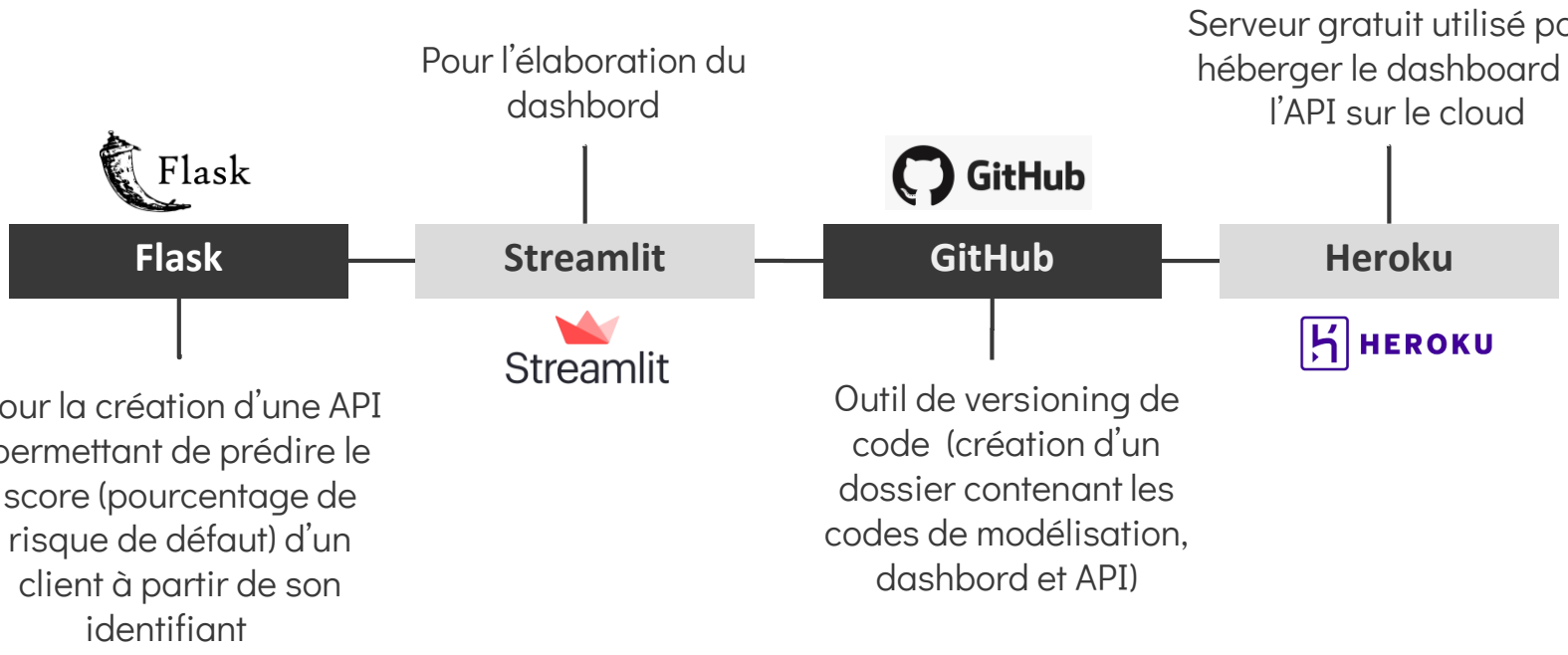
PRÉSENTATION DU DASHBOARD

Spécification attendues

Dashbord interactif permettant de :

- visualiser le score et l'interprétation de ce score pour chaque client de façon intelligible pour une personne non experte en data science.
- visualiser des informations descriptives relatives à un client (via un système de filtre)
- comparer les informations descriptives relatives à un client à l'ensemble des clients ou à un groupe de clients similaires

Technologies utilisées (déploiement)



DÉMO

https://p7-dashboard-mbiadou.herokuapp.com/

Conclusion

- Utilisation d'un Kernel Kaggle pour le feature engineering
- Entraînement et optimisation de plusieurs modèles via GridSearhCv
- Comparaison et sélection du meilleur modèle à l'aide du score AUC
- Définition d'une fonction coût adapté au métier
- Ré-optimisation des hyper-paramètres du modèle sélectionné (LGBM) avec HyperOpt en se basant sur la fonction coût métier
- Interprétabilité globale et locale du modèle LGBM avec SHAP
- Mise en production d'un Dashboard interactif avec Streamlit utilisant la prédiction faite par une API Flask.

MERCI

Des questions ?