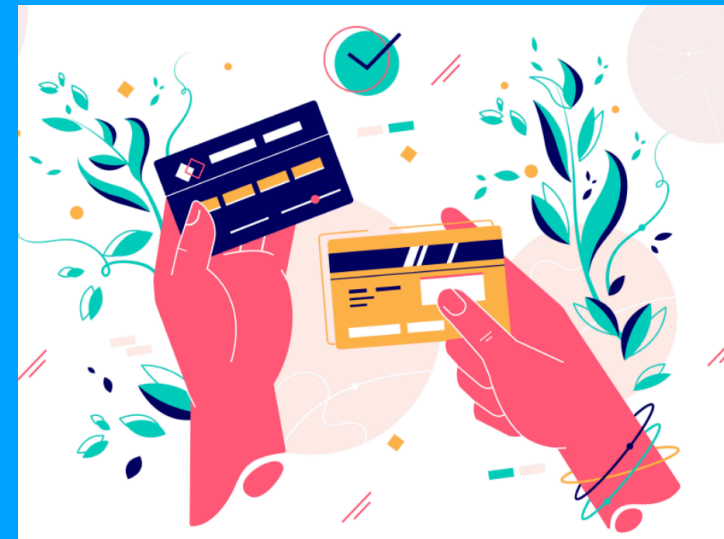


Implémentez un modèle de scoring

Prêt à dépenser

EC - octobre 2023



Contexte et problématique

Entreprise

- « Prêt à dépenser » est une société financière qui propose des **crédits à la consommation** à des personnes ayant peu ou pas d'historique de prêt.

Besoin

- La société souhaite mettre en oeuvre un **outil de « scoring crédit »** pour (i) calculer la probabilité de remboursement du credit et (ii) classier la demande (accordé ou refusé).

Mission

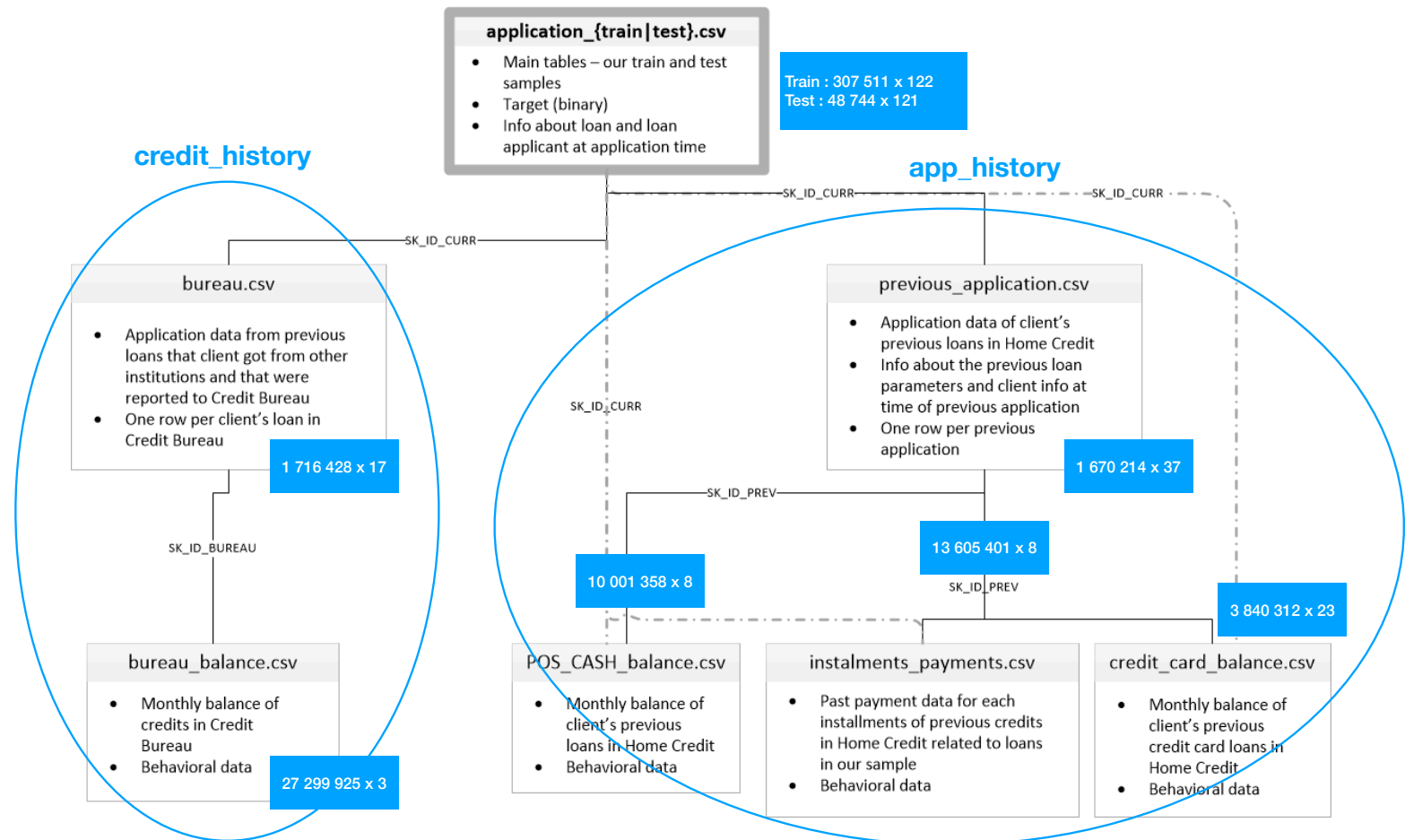
- Modéliser un **algorithme de classification** s'appuyant sur des sources de données variées (données comportementales, d'autres inst. Fi...),
- **Analyser les features** qui contribuent aux modèle,
- Mettre en production le modèle via une **API** et réaliser **une interface** de test de l'API



Présentation du jeu de données

- 8 jeux de données
- 217 features
- 307 511 crédits catégorisés (difficulté de remboursement ou non)
- 48 744 demandes de crédit à catégoriser (acceptée ou refusée)

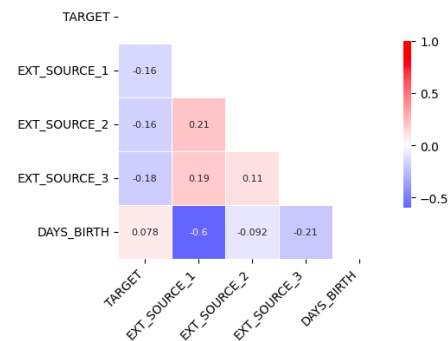
- **Application dataset** : variables relatives à la demande de crédit
- **credit_history** : variables relatives à des crédits accordés par d'autres organisations financières
- **app_history** : variables relatives à des crédits précédemment accordés par Prêt à Dépenser



Exploratory Data Analysis

1. Analyse application train

- Revue des valeurs incohérentes
- Traitement des **valeurs manquantes**
 - Suppression des features dont plus de la moitié des datas sont manquantes (sauf ext_source 1,2 et 3)
 - Variables catégorielles : imputation de la donnée la plus courante
 - Variables numériques : imputation de la valeur médiane
- Encoding des **variables catégorielles**
 - Label encoding pour les variables avec 2 catégories différentes
 - One hot encoding pour les autres variables
- **Création de variables** (8 variables)
 - INCOME_PER_PERSON
 - CREDIT_INCOME_RATIO
 - ...
- **Corrélation** avec la target



2. Analyse des crédits déjà accordés par le passé et des crédits accordées par d'autres instructions financières

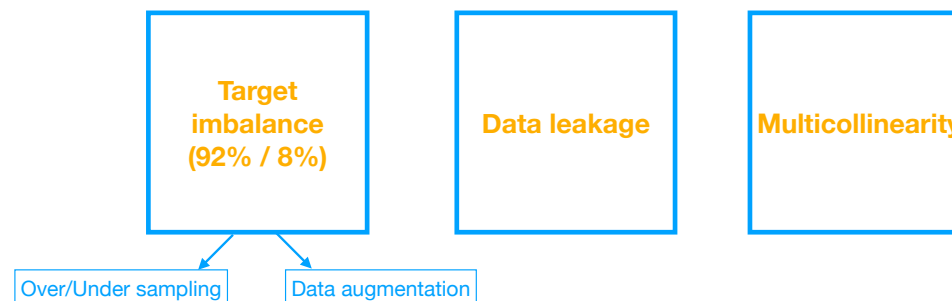
- Sélection de variables pertinentes via « groupby » (mean, max, min)
- Gestions des valeurs manquantes (idem train set)

3. Application test

- Application des mêmes modifications/creations que train set
- Gestion des valeurs manquantes & des variables catégorielles
- Vérification que la seule différence entre les deux sets est la target

4. Création de deux datasets « training » et « test » avec toutes les features (merge sur SK_ID_CURR avec credit history et app history)

- Les valeurs manquantes sont remplacées par 0
- Total final de **241 features** (sans compter la target)



Modélisation

Démarche de modélisation et choix des metrics

Scope : Classification binaire (2 classes) supervisée (les classes sont labellisées)

Data source de modélisation : application_train (241 features + target)
Split du dataset en train / test (70/30) :

- Comparaison des modèles sur la base des résultats obtenus sur le train set
- Détection de l'overfitting via le test set

Modèles comparés :

- Logistic Regression,
- Random Forest,
- XGBoost,
- LightGBM

Benchmark : Dummy Classifier

Pipeline

- ♦ **MinMax** scaler pour normaliser les variables,
- ♦ **SMOTE** pour gérer le déséquilibre des classes,
- ♦ **RandomizedSearchCV** pour optimiser les hyper-paramètres pertinents des différents algorithmes.

Indicateurs de performance

Metrics traditionnelles

- ❖ **Accuracy** : fréquence à laquelle le classificateur prédit correctement
★ Attention cas de déséquilibre de classe
- ❖ **AUC** (aire sous la courbe) : capacité du classificateur à faire la distinction entre les classes
- ❖ **Matrice de confusion** : tableau mettant en évidence les combinaisons de valeurs prédites vs. réelles (TP, TN, FP, FN)
- ❖ **Recall (sensitivity)** : combien de cas positifs réels sont prédit correctement par le modèle ($TP/(TP+FN)$)
★ Particulièrement pertinent dans les cas où les FN sont plus préoccupants que les FP

Score métier

- ❖ **Error Cost** : $10 \times FN + FP$
 - Score standardisé entre 0 et 1
 - Indiqué comme le score à optimiser dans la Random Search
- ❖ **Matrice de confusion des coûts** : tableau mettant en évidence les coûts générés en fonction du type et du nombre d'erreurs de prédiction

Modélisation

Visualisation du tracking via MLFlow UI

mlflow 2.7.1

Experiments

Models

[GitHub](#) [Docs](#)

Experiments

Search Experiments

☐ Default

☐ credit_scoring_4

☐ credit_scoring_3

☒ credit_scoring_2

☐ credit_scoring

credit_scoring_2

Provide Feedback

Share

Experiment ID: 323698739188100536

Artifact Location: mlflow-artifacts:/323698739188100536

> Description

Edit

metrics.rmse < 1 and params.model = "tree"

Time created

State: Active

Sort: Created

Columns

+ New run

Table

Chart

Evaluation

Experimental

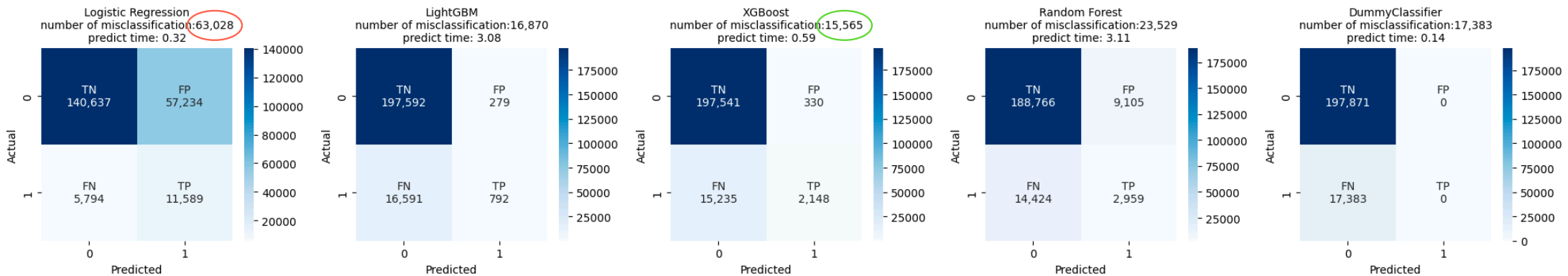
						Metrics		
<input type="checkbox"/>	<input type="radio"/>	Run Name	Created	Duration	Models	best_cv_score	eval_auc	eval_error_sco
<input type="checkbox"/>	<input type="radio"/>	<div>Best_XGB</div>	<div>2 days ago</div>	22.5s	<div>xgb_best/4, 1 more</div>	-	0.534	0.924
<input type="checkbox"/>	<input type="radio"/>	<div>XGB_finetuning</div>	<div>2 days ago</div>	12.2min	<div>sklearn, 1 more</div>	0.924	-	-
<input type="checkbox"/>	<input type="radio"/>	<div>Best_XGB</div>	<div>11 days ago</div>	20.5s	<div>xgb_best/1</div>	-	0.522	0.922
<input type="checkbox"/>	<input type="radio"/>	<div>Dummy Classifier</div>	<div>11 days ago</div>	4.9s	<div>sklearn</div>	-	0.5	0.919
<input type="checkbox"/>	<input type="radio"/>	<div>Random_forest</div>	<div>11 days ago</div>	29.3min	<div>sklearn, 1 more</div>	0.923	-	-
<input type="checkbox"/>	<input type="radio"/>	<div>Logistic Regression</div>	<div>11 days ago</div>	2.3h	<div>sklearn, 1 more</div>	0.925	-	-
<input type="checkbox"/>	<input type="radio"/>	<div>XGBoost</div>	<div>12 days ago</div>	20.4min	<div>sklearn, 1 more</div>	0.922	-	-
<input type="checkbox"/>	<input type="radio"/>	<div>LightGBM</div>	<div>12 days ago</div>	6.8min	<div>sklearn, 1 more</div>	0.921	-	-

Modélisation

Synthèse des résultats

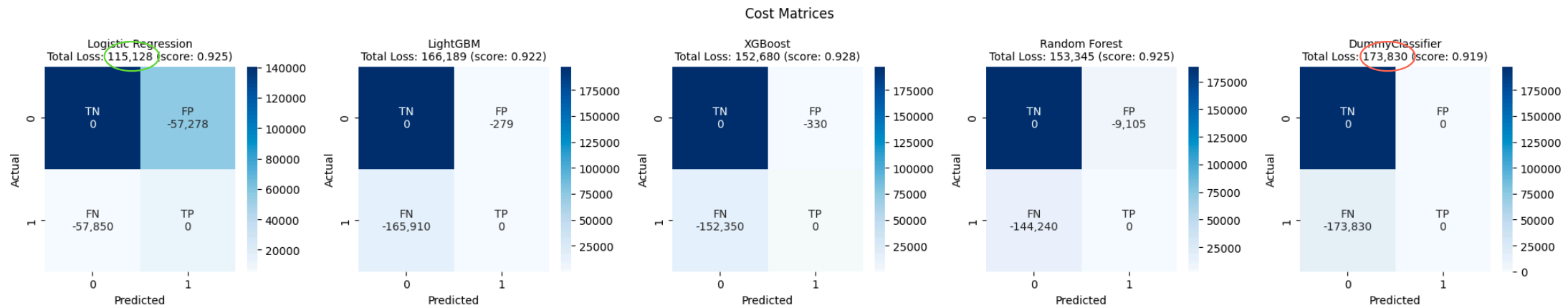
	Model	Error_cost_score	Error_cost	AUC	Recall	Accuracy	Mean_fit_time	Best_params
0	Logistic Regression	0.9246	23156.0	0.7515	0.6621	0.7077	141.72	{'classifier__solver': 'saga', 'classifier__pe...
1	LightGBM	0.9210	29606.0	0.7673	0.3773	0.9192	13.81	{'classifier__num_leaves': 70, 'classifier__ma...
3	Random Forest	0.9228	30322.6	0.6976	0.2555	0.8875	64.16	{'classifier__n_estimators': 150, 'classifier__...
2	XGBoost	0.9223	31657.4	0.7638	0.2027	0.9190	46.94	{'classifier__subsample': 0.7, 'classifier__mi...
4	Dummy Classifier	0.9192	173830.0	0.5000	0.0000	0.9192	32.00	most_frequent

Confusion Matrices

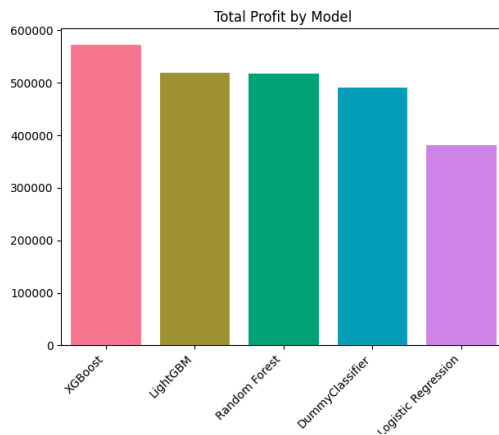


Modélisation

Choix du meilleur modèle



$$\text{Profit} = 6 \times \text{TN} - 4 \times \text{FP} - 40 \times \text{FN} + 0 \times \text{TP}$$



XGBoost génère 574 526 \$

- ➔ Identification d'un **nombre important de TN** (197 541, soit 99.8% des clients à identifier comme ne faisant pas défaut),
- ➔ Tout en **limitant le nombre d'erreurs total** (15 565 vs. 63 028 pour LogReg)

Logistic Regression génère 383 046 \$ (-33% vs. XGBoost)

- ➔ **Minimisation du nombre de FN** (5 794 vs. 15 235 pour XGBoost) et donc des cout associés,
- ➔ **Au détriment du nombre de TN** (71% des clients à identifier comme ne faisant pas défaut) et de **FP** (manque à gagner)

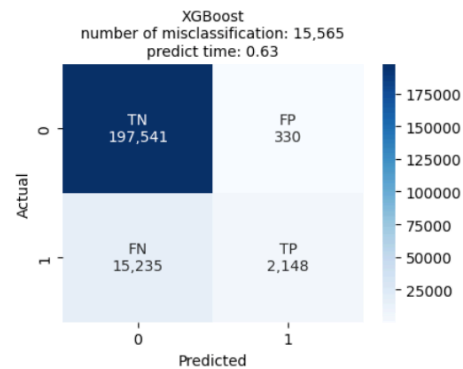
Modélisation

Finetuning du meilleur modèle

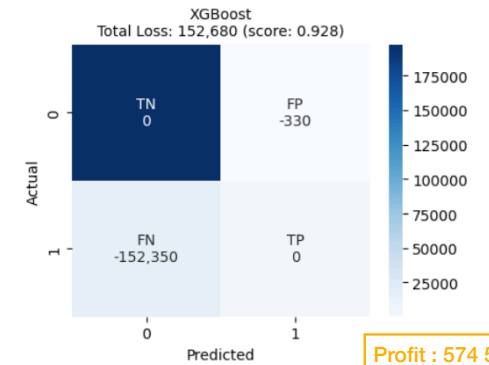
Meilleur modèle issu de l'optimisation des hyperparamètres initial

```
param_classifier__max_depth    60
param_classifier__min_child_weight  50
param_classifier__subsample    0.7
param_classifier__eta          0.1
```

Confusion Matrices



Cost Matrices



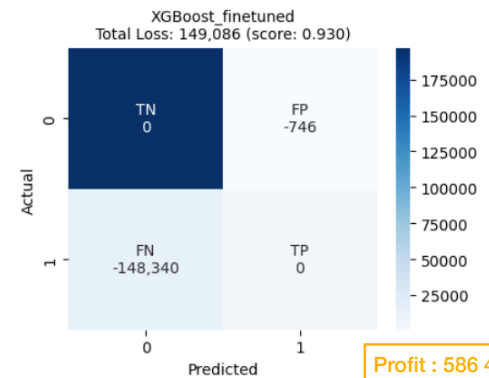
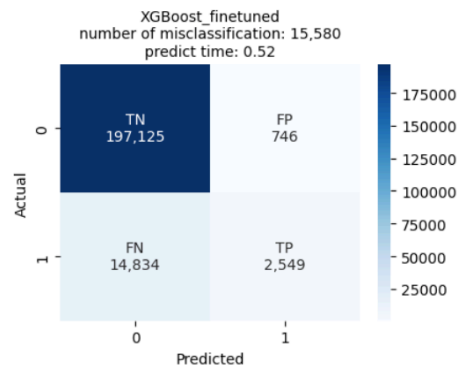
Profit : 574 526 \$



Modèle finetuné

```
param_classifier__max_depth    50
param_classifier__min_child_weight  70
param_classifier__subsample    0.5
param_classifier__eta          0.2
```

Nombre d'erreur légèrement plus important
mais mieux réparties : plus de FP (et de TP) au
détriment des FN (plus couteuses)

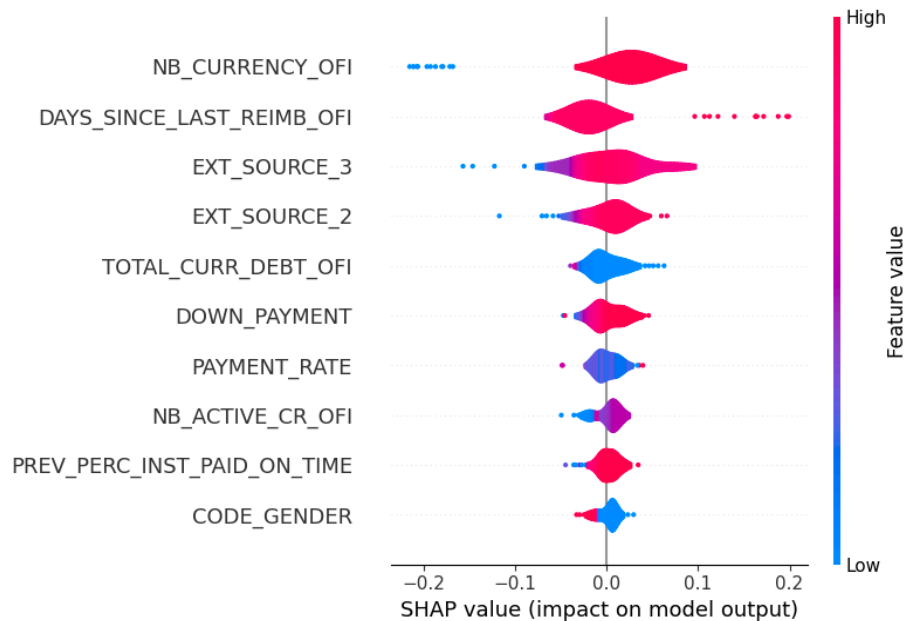


Profit : 586 406 \$

Modélisation

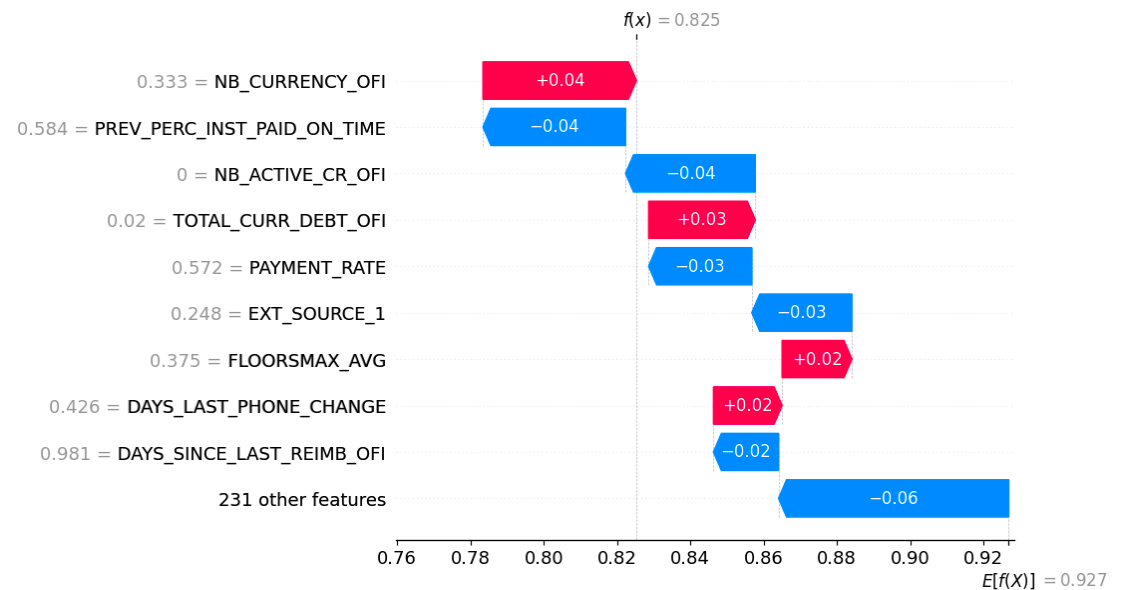
Analyse des features les plus importantes

Top 10 des features qui **contribuent de manière globale** à la prédiction en classe 0 (pas de défaut de paiement)



Explication globale

Principales features expliquant la probabilité de remboursement (82.5%) d'un individu en particulier

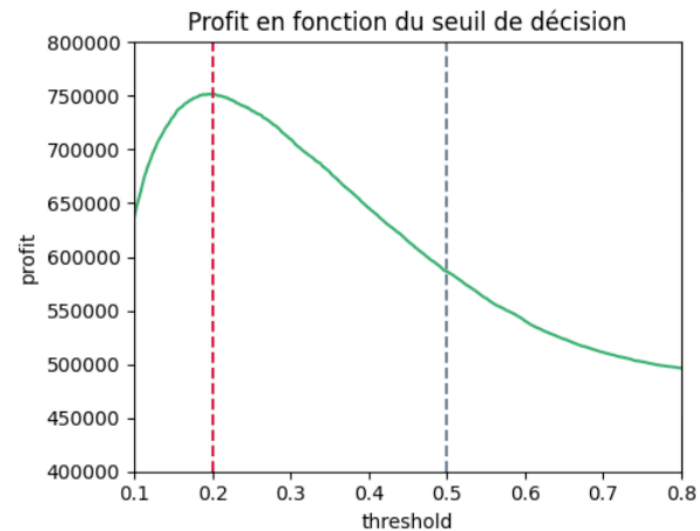
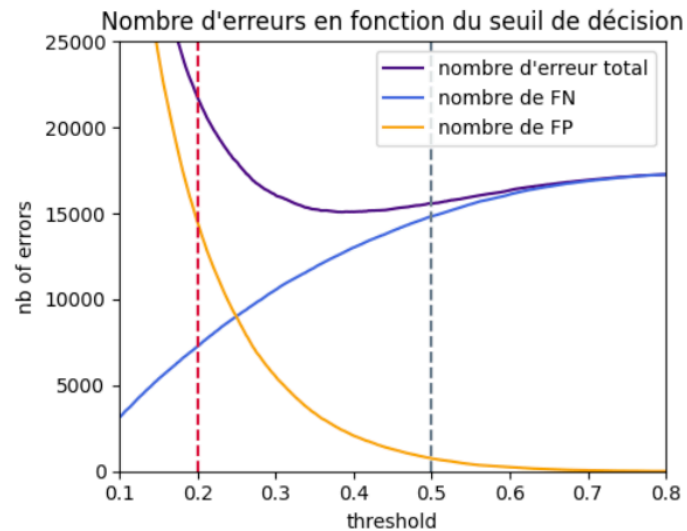


Explication locale

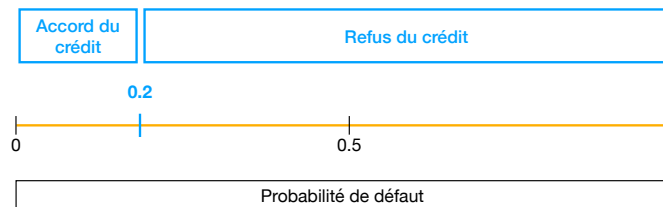
Modélisation

Optimisation du seuil (probabilité de défaut)

Objectif : définir le seuil de décision permettant de maximiser le profit

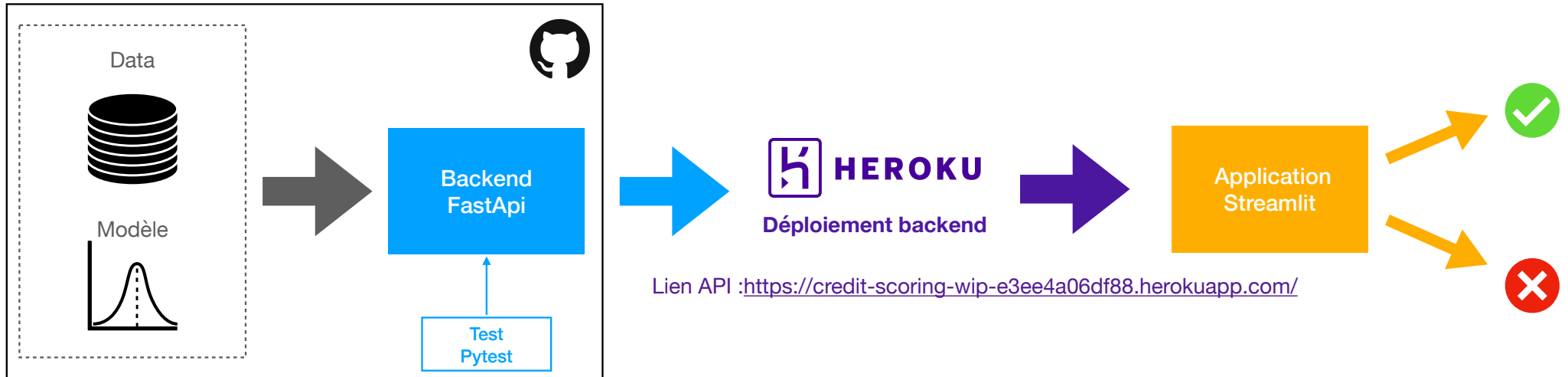


Seuil optimal : 0.20
Profit : 751 976



Pipeline de déploiement

Lien Github : https://github.com/estellec18/app_credit_scoring



La robustesse et l'efficacité de FastAPI pour le traitement backend, la gestion des données et de la logique business

La simplicité et l'interactivité de Streamlit pour les interfaces utilisateurs et la visualisation des données

Exemple d'un scoring client via API/Streamlit

Selection du client

Veuillez spécifier le numéro d'identification du demandeur de prêt

208550

Genre: Male

Situation familiale: Married

Nombre d'enfants: 2

Montant du crédit demandé: 854,896

Revenu: 450,000

Source du revenu: Working

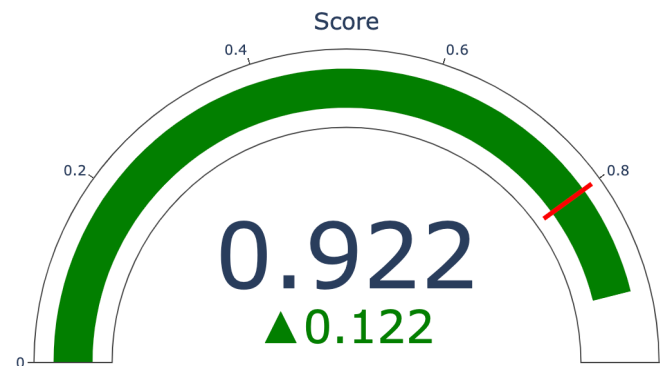
Prédiction de la capacité de remboursement d'un demandeur de prêt

! Cet outil permet d'assister à la prise de décision et doit être utilisé conjointement avec une analyse approfondie réalisée par un professionnel !

Prediction

Demande de crédit acceptée ✓

Nous estimons la probabilité de default du client à : 7.80%



Analyse de data drift

Par défaut :

- Le data drift est détecté si **au moins 50%** des variables ont drifté
- Le seuil de drift d'une feature est à **0.1**

Les algorithmes utilisés par evidently pour les datasets avec plus de 1000 observations

- **Wasserstein distance** pour les features numériques
- **Jensen-Shannon divergence** pour les features catégorielles

Dataset Drift

Dataset Drift is NOT detected. Dataset drift detection threshold is 0.5

241

Columns

23

Drifted Columns

0.0954

Share of Drifted Columns

Data Drift Summary

Drift is detected for 9.544% of columns (23 out of 241).

Search						
Column	Type	Reference Distribution	Current Distribution	Data Drift	Stat Test	Drift Score
> prediction	cat			Detected	Jensen-Shannon distance	0.166365
> PAYMENT_RATE	num			Detected	Wasserstein distance (normed)	0.586121
> AMT_REQ_CREDIT_BUREAU_QRT	num			Detected	Wasserstein distance (normed)	0.426334
> CREDIT_INCOME_RATIO	num			Detected	Wasserstein distance (normed)	0.310787
> AMT_REQ_CREDIT_BUREAU_MON	num			Detected	Wasserstein distance (normed)	0.258638

Limites

Ce projet constitue un « proof of concept ».

La pertinence des résultats obtenus repose sur un certains nombres d'hypothèses qui nécessitent d'être validées par un « expert métier » :

- * Approximation des calculs de coûts et de profits relatifs aux emprunts accordés / refusés / non remboursés
- * Sélection et traitement des variables à affiner










Annexes

Bibliothèques principales








Les travaux ont été réalisés sur Python (3.9.6)

- Bibliothèques générales
 - numpy 1.23.5
 - pandas 2.1.0
 - matplotlib 3.7.3
 - seaborn 0.12.2
- Machine learning et MLOps:
 - scikit-learn 1.3.1
 - xgboost 2.0.0
 - lightgbm 4.1.0
 - shap 0.42.1
 - joblib 1.3.2
- API :
 - fastapi 0.103.2
 - pytest 7.4.2
 - plotly 5.17.0
 - streamlit 1.27.1
- Data drift :
 - evidently 0.4.5

Commit history

maj du sommaire	...
 estellec18 pushed 1 commit to main · 860012b...86387a4 · 3 minutes ago	
mise à jour du sommaire	...
 estellec18 pushed 1 commit to main · 11428fa...860012b · 21 minutes ago	
update des links	...
 estellec18 pushed 1 commit to main · d06a590...11428fa · 45 minutes ago	
update des links	...
 estellec18 pushed 1 commit to main · 7e9241d...d06a590 · 46 minutes ago	
update des links	...
 estellec18 pushed 1 commit to main · 9cf722c...7e9241d · 47 minutes ago	
last version du modèle	...
 estellec18 pushed 1 commit to main · 7408848...9cf722c · 51 minutes ago	
data drift avec dernier modèle	...
 estellec18 pushed 1 commit to main · 9ab788b...7408848 · 53 minutes ago	



update EDA et upload de modélisation	...
 estellec18 pushed 1 commit to main · 140e0ba...fa82df6 · 16 days ago	
delete unnecessary files	...
 estellec18 pushed 2 commits to main · e07631b...140e0ba · 17 days ago	
Update README.md	...
 estellec18 pushed 1 commit to main · 683313a...e07631b · 18 days ago	
Delete 1. EDA_v3.ipynb	...
 estellec18 pushed 1 commit to main · fa4a88d...683313a · 18 days ago	
modif nom du fichier	...
 estellec18 pushed 1 commit to main · 2f369aa...fa4a88d · 18 days ago	
première version EDA	...
 estellec18 pushed 1 commit to main · e95d69e...2f369aa · 19 days ago	
Initial commit	...
 estellec18 created main · e95d69e · 19 days ago	

Lien vers le **repository principal** :
https://github.com/estellec18/modele_de_scoring

Lien vers le **repository de l’api** :
https://github.com/estellec18/app_credit_scoring

Pytest

Résultats des tests réalisés en local

```
(env) estellecampos@Air-de-Estelle app_credit_scoring % pytest -v
===== test session starts =====
platform darwin -- Python 3.9.6, pytest-7.4.2, pluggy-1.3.0 -- /Users/estellecampos/Documents/Reconversion/Formation
OpenClassroom/Projet_7_bis/env/bin/python3
cachedir: .pytest_cache
rootdir: /Users/estellecampos/Documents/Reconversion/Formation OpenClassroom/Projet_7_bis/app_credit_scoring
plugins: anyio-3.7.1
collected 12 items

test_main.py::test_not_empty PASSED [ 8%]
test_main.py::test_duplicate_in_data PASSED [ 16%]
test_main.py::test_numclient_in_data PASSED [ 25%]
test_main.py::test_age PASSED [ 33%]
test_main.py::test_credit PASSED [ 41%]
test_main.py::test_can_call_endpoint PASSED [ 50%]
test_main.py::test_predict_valid PASSED [ 58%]
test_main.py::test_predict_default PASSED [ 66%]
test_main.py::test_gauge PASSED [ 75%]
test_main.py::test_update_gauge PASSED [ 83%]
test_main.py::test_explanation PASSED [ 91%]
test_main.py::test_perso_info PASSED [100%]

===== 12 passed in 3.74s =====
```

Résultat des tests réalisés via Github Actions

```
> Set up job 1s
> Run actions/checkout@v3 2s
> Set up Python 3.9.6 8s
> Install dependencies 52s
v Test with pytest 10s

1 ▶ Run pytest test_main.py
7 ===== test session starts =====
8 platform linux -- Python 3.9.6, pytest-7.4.2, pluggy-1.3.0
9 rootdir: /home/runner/work/app_credit_scoring/app_credit_scoring
10 plugins: anyio-3.7.1
11 collected 12 items
12
13 test_main.py ..... [100%]
14
15 ===== 12 passed in 0.18s =====
```