



Projet P7

Implémentation d'un modèle de scoring

Introduction

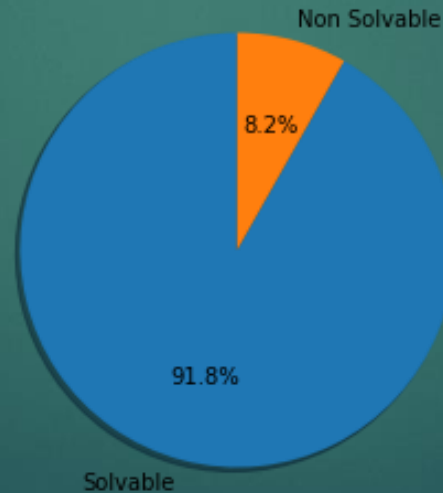
- DataScientist pour Prêt à dépenser -> propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt
- **Objectifs :**
 - Développement d'un outil de "scoring crédit" pour calculer la probabilité qu'un client rembourse son crédit
 - Développer un algorithme de classification
 - Développer un dashboard interactif
 - Déployer sur le web le dashboard et l'API de prédiction



Données

9 fichiers :

- 307 511 clients pour 122 Indicateurs
- Client : informations générales + informations sur les prêts précédents
- Fort déséquilibre entre les personnes solvables et les personnes non solvables.



Sommaire



1. Pré-Traitement & Analyse de données
2. Recherche d'un modèle optimale
3. API
4. Dashboard
5. Conclusion

Pré-Processing

- Utilisation du kernel *lightgbm_with_simple_features.py* récupérer sur kaggle
 - Taille dataset : 307488 lignes x 797 colonnes
- Vérification des valeurs manquantes
 - On supprime les lignes avec + 30% de valeurs manquantes
 - On supprime les colonnes avec + 57% de valeurs manquantes
 - Taille dataset : 290970 lignes x 577 colonnes
- Split le jeux de donnée
- Imputation
- Normalisation

Recherche d'un modèle optimale

- **Comparaison de modèles :**
 - KNN Classifier
 - Random Forest
 - Light-GBM (LGBM Classifier)
- **Développement d'une métrique bancaire**
- **Gérer le déséquilibre de classes**
- **Recherche Hyperparamètres**

Recherche d'un modèle optimale

Développement d'une métrique bancaire

Confusion matrix		Reality	
		Negative : 0	Positive : 1
Prediction	Negative : 0	True Negative : TN	False Negative : FN
	Positive : 1	False Positive : FP	True Positive : TP

TN : client capable de rembourser = négatif au refus (classe 0)

TP : client incapable de rembourser = positif au refus (classe 1)

Recherche d'un modèle optimale

Développement d'une métrique bancaire

Confusion matrix		Reality	
		Negative : 0	Positive : 1
Prediction	Negative : 0	True Negative : TN	False Negative : FN
	Positive : 1	False Positive : FP	True Positive : TP

FN : Accorder un crédit à un client incapable de rembourser = PERTE

TN : Accorder un crédit à un client capable de payer = GAIN

TP : Pas accorder de prêt à un client incapable de rembourser = NUL

FP : Pas accorder de prêt à un client capable de rembourser = PERTE

Recherche d'un modèle optimale

Développement d'une métrique bancaire

Notre intérêt est de gagner de l'argent

Donc on va attribuer des coefficients à nos différents paramètres :

- Coefficient Négatif : FN et FP
- Coefficient Positif : TN

Attribuer un crédit à une personne incapable = impact plus fort sur le score

- Coeff_FN = -100
- Coeff_FP = -1
- Coeff_TN = 10
- Coeff_TP = 0

$$\text{Scoring} = 10 \times TN + 0 \times TP + (-100) \times FN + (-1) \times FP$$

$$\text{Scoring normalisé} = \frac{\text{Scoring} - \min_{\text{gain}}}{\max_{\text{gain}} - \min_{\text{gain}}}$$

Recherche d'un modèle optimale

Autres Métriques pour tester l'efficacité

$$\text{Recall}_1 = \frac{TP}{TP+FN} ; \text{Precision}_1 = \frac{TP}{TP+FP} ; \text{F1score}_1 = \frac{2 \times \text{precision}_1 \times \text{recall}_1}{\text{precision}_1 + \text{recall}_1}$$

$$\text{Recall}_0 = \frac{TN}{TN+FP} ; \text{Precision}_0 = \frac{TN}{TN+FN} ; \text{F1score}_0 = \frac{2 \times \text{precision}_0 \times \text{recall}_0}{\text{precision}_0 + \text{recall}_0}$$

Scoring_normalisé

On cherche à maximiser toutes ces métriques

Recherche d'un modèle optimale

Test des modèles

Recherche hyperparamètres avant de régler le problème de classe car **problèmes de performance de l'ordinateur**

Tableau comparatifs des différents modèles :

	TN	FP	FN	TP	precision_1	recall_1	f1-score_1	precision_0	recall_0	f1-score_0	scoring
LGBM & without Sampling	87769	248	7739	265	0.516569	0.033108	0.062228	0.918970	0.997182	0.956480	0.560877
KNN & without Sampling	88010	7	7998	6	0.461538	0.000750	0.001497	0.916694	0.999920	0.956500	0.547731
Random Forest & Without Sampling	88017	0	8004	0	0.000000	0.000000	0.000000	0.916643	1.000000	0.956509	0.547435

LGBM nous donne des performance légèrement meilleurs aux autres modèles

Mais le déséquilibres n'est pas géré

Recherche d'un modèle optimale

Déséquilibre de Classe

- Over-sampling : dupliquer aléatoirement des données existantes de la classe sous-représentée
- SMOTE : créer de nouvelles données à partir des données déjà existantes pour la classe sous-représentée
- Under-sampling : sélectionner des données de la classe sur-représentée

Ces méthodes sont ensuite utilisé pour entrainés nos modèles

Recherche d'un modèle optimale

Test des modèles

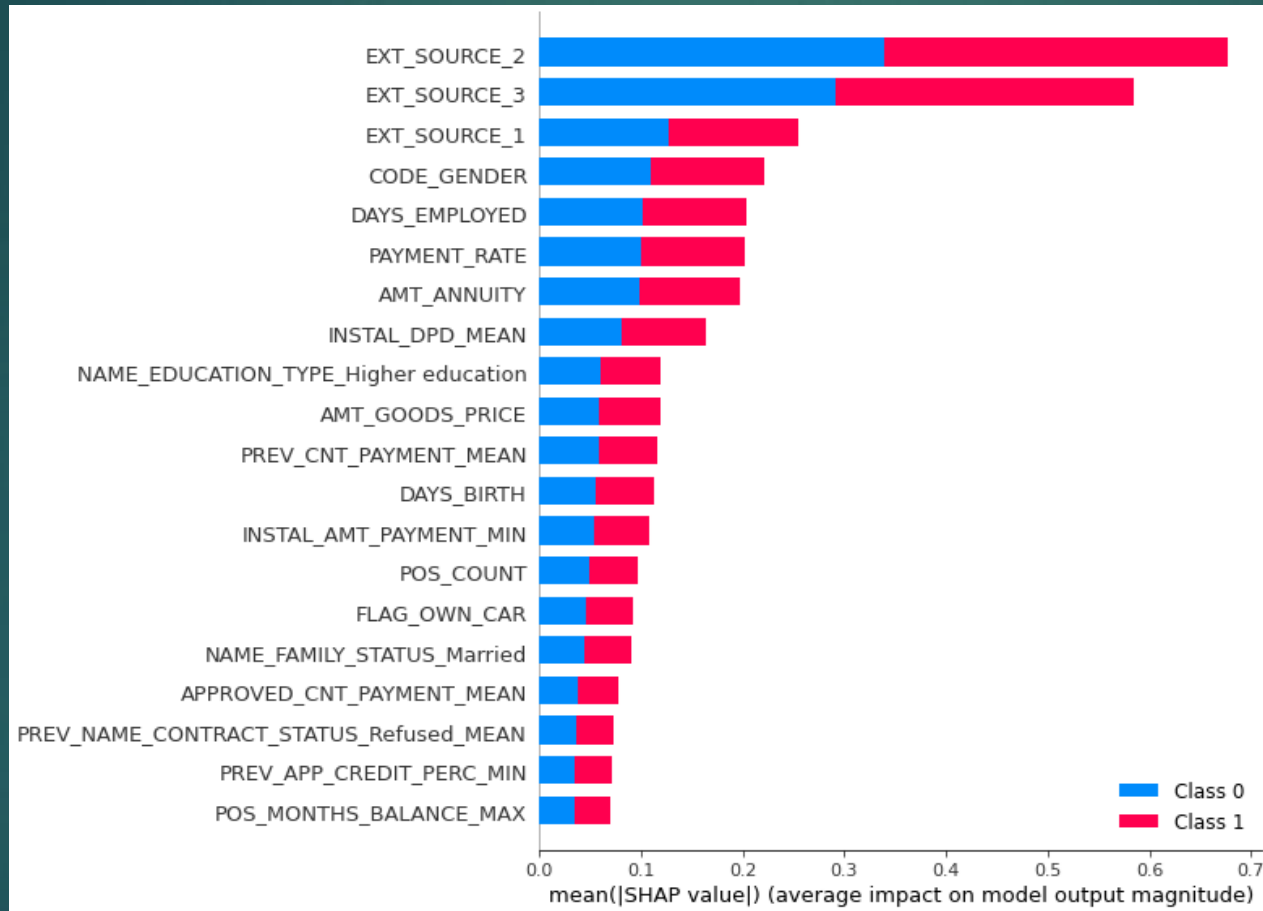
	TN	FP	FN	TP	precision_1	recall_1	f1-score_1	precision_0	recall_0	f1-score_0	scoring
LGBM Classifier & Over Sampling	65186	22831	2610	5394	0.191107	0.673913	0.297773	0.961502	0.740607	0.836721	0.710424
LGBM Classifier & Under Sampling	62085	25932	2336	5668	0.179367	0.708146	0.286234	0.963739	0.705375	0.814561	0.706629
Random forest Classifier & Over Sampling	52106	35911	2810	5194	0.126359	0.648926	0.211529	0.948831	0.591999	0.729097	0.617762
Random forest Classifier & Under Sampling	51349	36668	2775	5229	0.124806	0.653298	0.209575	0.948729	0.583399	0.722508	0.615033
KNN Classifier & Under Sampling	58168	29849	3595	4409	0.128700	0.550850	0.208651	0.941794	0.660872	0.776713	0.611080
Random forest Classifier & SMOTE	64908	23109	4791	3213	0.122065	0.401424	0.187205	0.931262	0.737448	0.823100	0.585376
KNN Classifier & Over Sampling	55211	32806	4000	4004	0.108775	0.500250	0.178694	0.932445	0.627277	0.750007	0.569789
LGBM Classifier & SMOTE	87801	216	7793	211	0.494145	0.026362	0.050053	0.918478	0.997546	0.956381	0.558022
KNN Classifier & SMOTE	21153	66864	1017	6987	0.094609	0.872939	0.170717	0.954127	0.240329	0.383947	0.526625

LGBM avec un OVERSAMPLING nous donne les meilleurs performances

On refait une recherche d'hyperparamètres uniquement pour LGBM & OVERSAMPLING

Recherche d'un modèle optimale

Features Importance

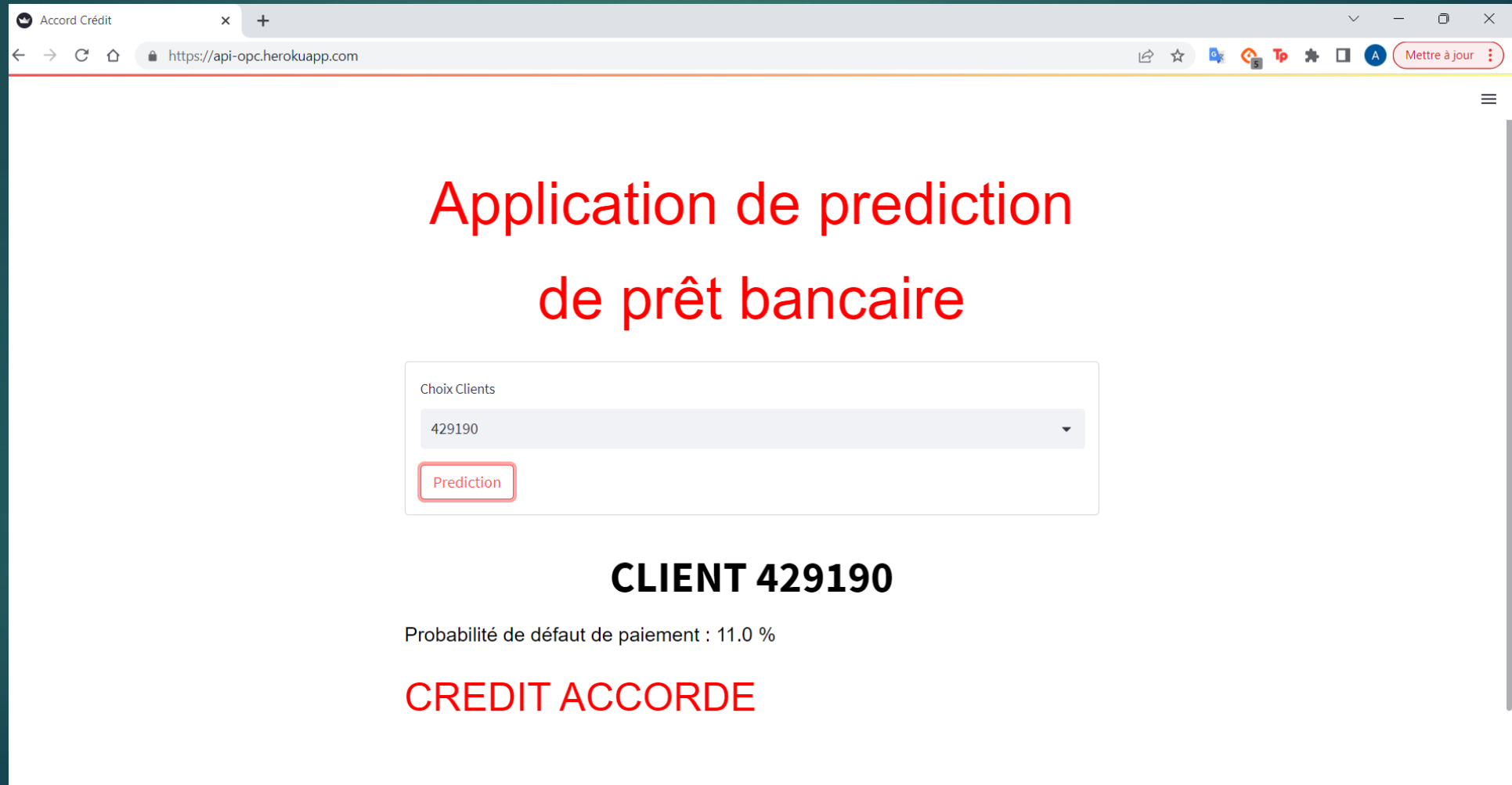


EXT_SOURCE : sources normalisées créées à partir de sources de données externes

Api

Lien : <https://aginth02-p7-opc-api-api-lwnypu.streamlit.app/>

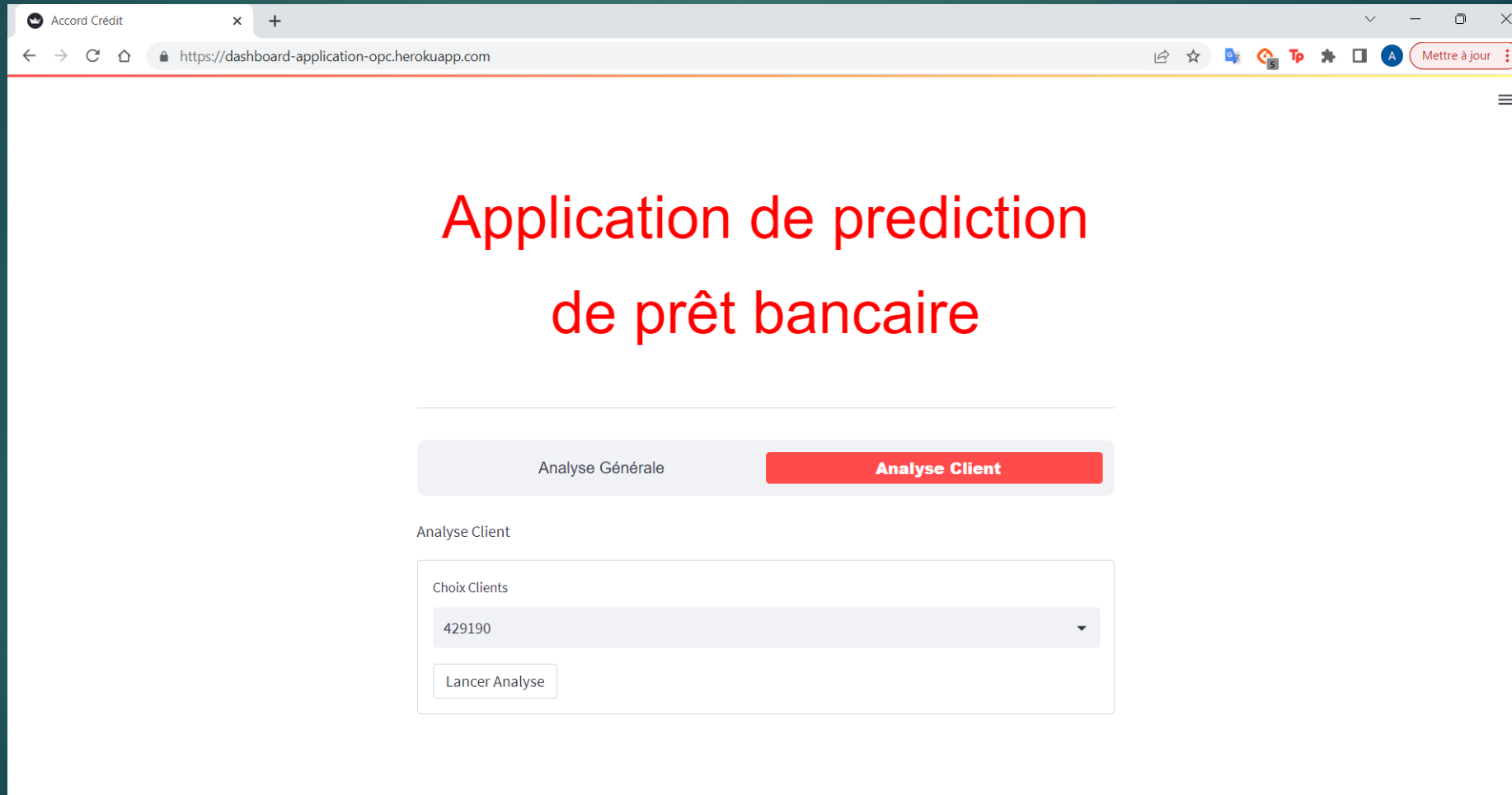
Lien github : <https://github.com/Aginth02/P7 OPC API>



The screenshot shows a web browser window with the address bar displaying `https://api-opc.herokuapp.com`. The page content is as follows:

- Header: "Accord Crédit" with a close button and a plus sign.
- Navigation bar: Back, forward, refresh, home, and address bar icons. A "Mettre à jour" button is on the right.
- Main content area:
 - Section header: "Application de prediction de prêt bancaire" in red.
 - Form:
 - Label: "Choix Clients".
 - Dropdown menu: Shows "429190".
 - Button: "Prediction" (highlighted with a red border).
 - Result: "CLIENT 429190" in bold black text.
 - Text: "Probabilité de défaut de paiement : 11.0 %".
 - Final result: "CREDIT ACCORDE" in red.

Lien github : https://github.com/Aginth02/P7_OPENCLASSROOM



Conclusion

- Définir les coefficients optimaux pour notre métrique
- Rendre le modèle plus éthique :
 - Enlevés les variables discriminantes
 - Comparaison modèles avec et sans ces variables
 - Perte de rentabilité pour la banque ?
- Créer de nouvelles variables
- Adaptation du Dashboard aux souhaits de la banque



MERCI DE VOTRE ATTENTION