



10/02/2024

# Implémentez un modèle de scoring

Armand FAUGERE [LinkedIn](#)

armand-faugere@live.fr

# Sommaire

- I) Cadrage du projet et données d'entrée
- II) Analyse exploratoire
- III) Modélisation
- IV) Réalisation de l'API et déploiement
- V) Analyse du data drift
- VI) Conclusion



# I) Cadrage du projet et données d'entrée

# I) Cadrage du projet et données d'entrée



## ❑ Contexte :

- Projet de mise en œuvre d'un outil de «scoring credit»
- ➔ probabilité qu'un client rembourse son crédit
- ➔ classification de la demande en « accordée » ou « refusée ».
- Proposition de crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt
- ➔ Risque financier important pour l'entreprise Prêt à dépenser

## ❑ But :

- Développer et mettre en production un modèle de scoring pour prédire la probabilité de faillite d'un client, et sa classification en « refusé » ou « accepté », avec une démarche MLOPS

## ❑ Objectifs :

- Réaliser une analyse exploratoire et sélectionner les variables pertinentes
- Modéliser, optimiser et choisir le meilleur modèle
- Analyser l'impact des variables retenues
- Réaliser une API et la déployer avec les tests associés
- Analyser en production le datadrift

## ❑ Le jeu de données ➔ 8 fichiers csv + 1 de description

- application\_train.csv  
➔ 122 colonnes, 307511 lignes
- application\_test.csv  
➔ 121 colonnes, 48744 lignes
- bureau\_balance.csv  
➔ 17 colonnes, 1716428 lignes
- bureau.csv  
➔ 3 colonnes, 27299924 lignes
- POS\_CASH\_balance.csv  
➔ 8 colonnes, 10001357 lignes
- instalments\_payments.csv  
➔ 8 colonnes, 13605400 lignes
- credit\_card\_balance.csv  
➔ 23 colonnes, 3840311 lignes
- previous\_application.csv  
➔ 37 colonnes, 1670213 lignes

**Principes de protection des données** (finalité, proportionnalité et pertinence, durée de conservation limitée, sécurité et confidentialité, droits des personnes)  
[www.cnil.fr](http://www.cnil.fr)



## II) Analyse exploratoire

## 2) Analyse exploratoire



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn

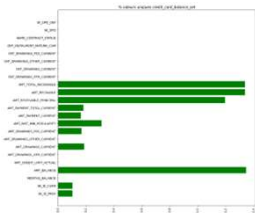
Analyse des jeux de données

Fusion/ Agrégation jeux de données

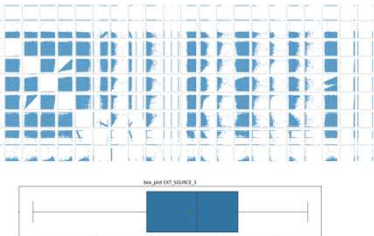
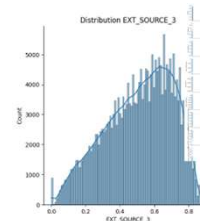
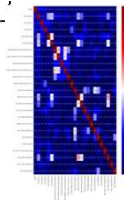
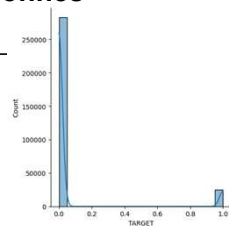
Feature Ingeneering

Features Selection

- ❑ Analyse info sur datasets (nb de lignes, colonnes...)
- ❑ Analyse valeurs nulles, valeurs uniques et doublons

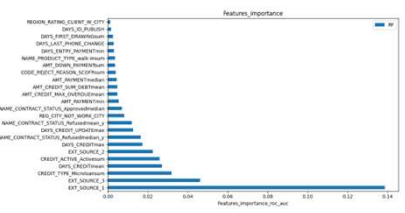


- ❑ Fonction agrégation num/categ → ["count", "mean", "max", "min", "sum", "median"]
- ❑ Agrégation → SK\_ID\_BUREAU, SK\_ID\_CURR
- ❑ Analyse des jeux de données fusionnés train & test → **1588 colonnes**



- ❑ Corrélation de spearman filtre > 0,05
- ❑ Traitement des valeurs nulles → Suppression des colonnes nan > 60%
- ❑ Traitement des multicollinéarités → Suppression des colonnes corrélées > 60%
- ❑ Imputation valeurs manquantes itérative imputer
- ❑ Analyse univariée et bvariée (displot, boxplot, pairplot)

- ❑ Analyse Feature importance globale des **24 colonnes** → On conserve les colonnes
- ❑ Filtre sur Train et Test avec les features sélectionnées

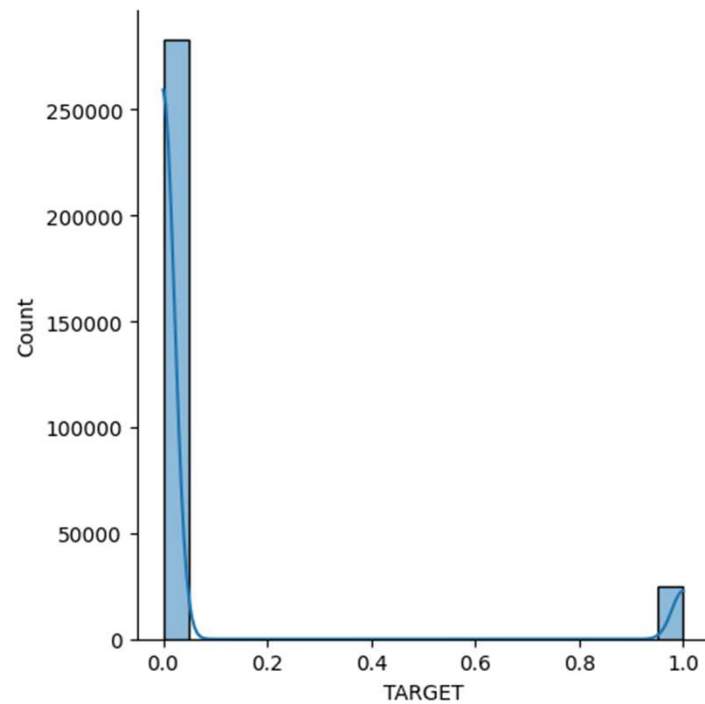


1  
Exploration  
Data  
Analysis

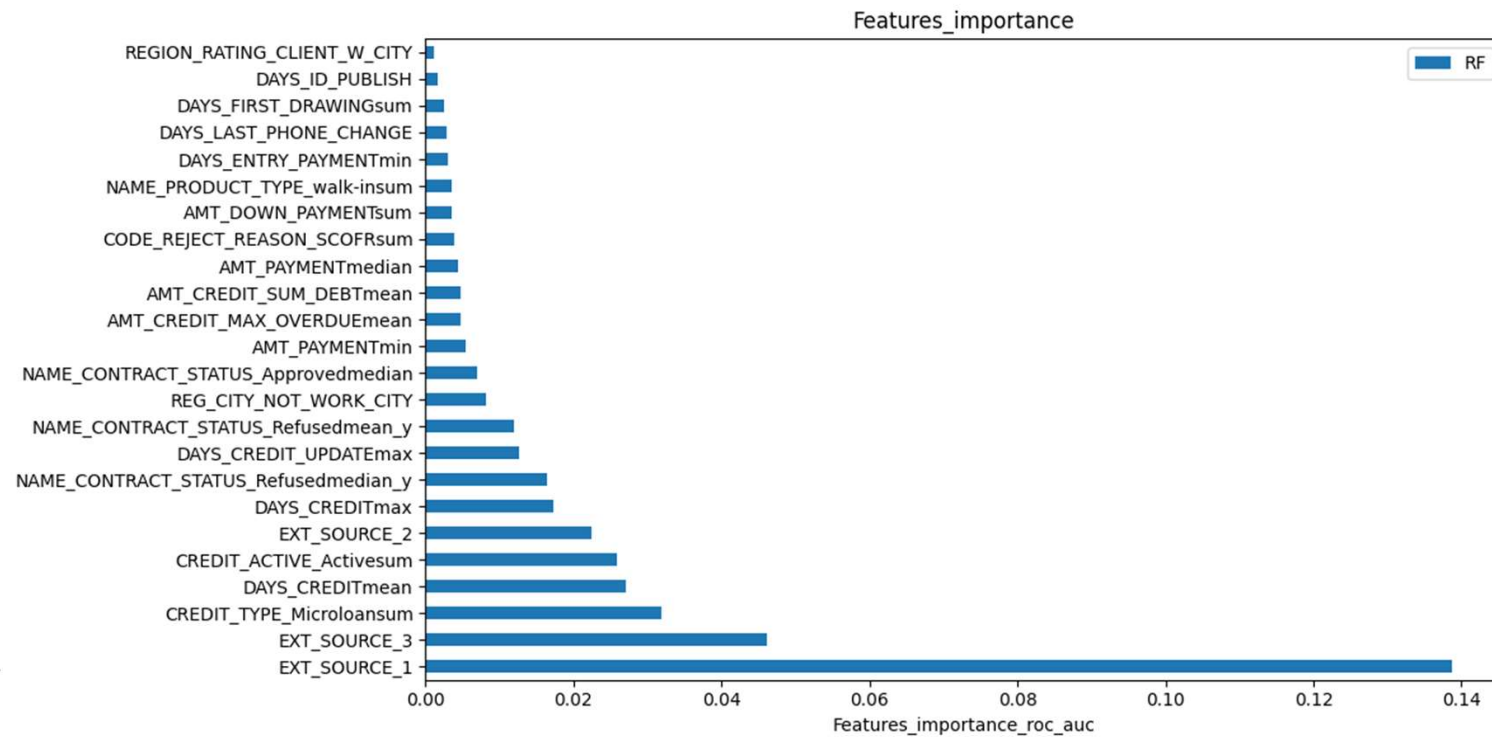
## 2) Analyse exploratoire



Distribution de la TARGET



Feature importance des variables retenues (RandomForestClassifier)



# III) Modélisation



# 3) Modélisation



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn, imblearn, time, mlflow, lightgbm, shap, joblib

2

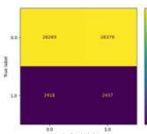
Modélisation  
Essais  
Tracking

Préparation

- ☐ Découpage set de train (train + test)
- ☐ Métriques :
  - accuracy score
  - roc auc score
  - recall score
  - confusion matrix
  - f1 score
- ☐ Algorithmes :
  - dummy classifier
  - logistic regression
  - SVC
  - random forest classifier
  - adaboostclassifier
  - lgmbc classifier
  - decision tree classifier

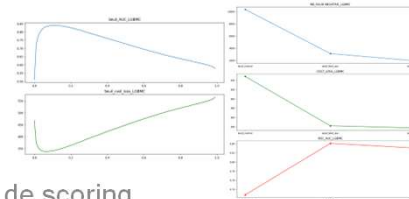
Optimisation  
Paramétrique  
auc\_score

- ☐ Hypers paramètres
- ☐ Sélection des algorithmes :
  - dummy classifier
  - logistic regression
  - lgmbc classifier
  - decision tree classifier
- ☐ Tests auc\_score
- ☐ Tracking via mlflow (accuracy, auc, durée)



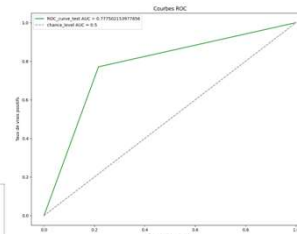
Optimisation  
Métier  
Hypers paramètres,  
Tests, Tracking

- ☐ Hypers paramètres
- ☐ Tests métriques customisées :
  - betascore
  - recall
  - spécificité
- ☐ Tests fonction de cout
- ☐ Optimisation des seuils
- ☐ Tracking via mlflow (accuracy, auc, durée, cost\_loss)



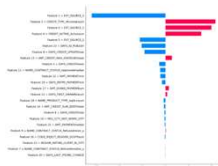
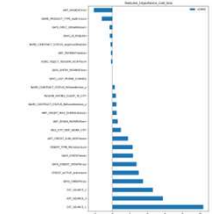
Choix du  
modèle

- ☐ Sélection meilleur modèle et hypers paramètres
- auc\_score
- cost\_loss
- LGMBC**



Interprétation

- ☐ globale
- ☐ locale



### 3) Modélisations



Optimisation hyper paramètres refit → roc\_auc

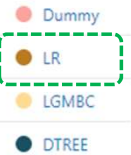
Algorithme	Statut	auc_score
dummy classifier	Retenu	0.500488
logistic regression	Retenu	0.781102
SVC	Non retenu (pas de convergence)	–
random forest classifier	Non retenu (pas de convergence)	–
adaboostclassifier	Non retenu (pas de convergence)	–
lgmbc classifier	Retenu	0.870352
decision tree classifier	Retenu	0.699927

Cette première itération d'optimisation paramétrique a permis de **sélectionner les algorithmes à utiliser pour les étapes suivantes**

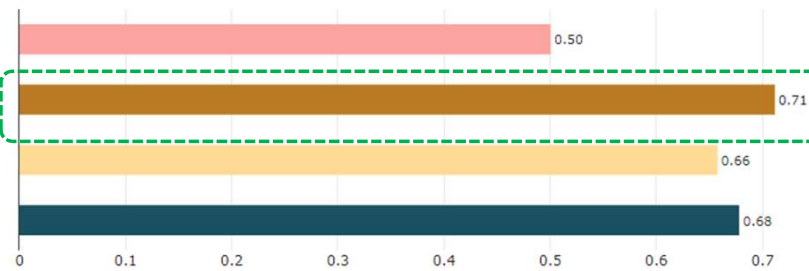
### 3) Modélisations



Expérimentations/Tests meilleurs hypers paramètres → roc\_auc



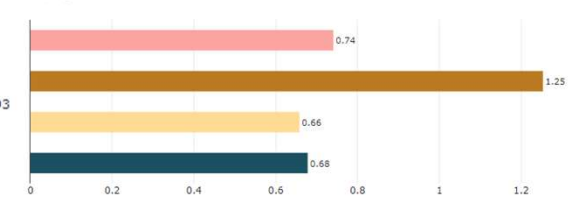
roc\_auc  
Comparing first 4 runs



accuracy  
Comparing first 4 runs

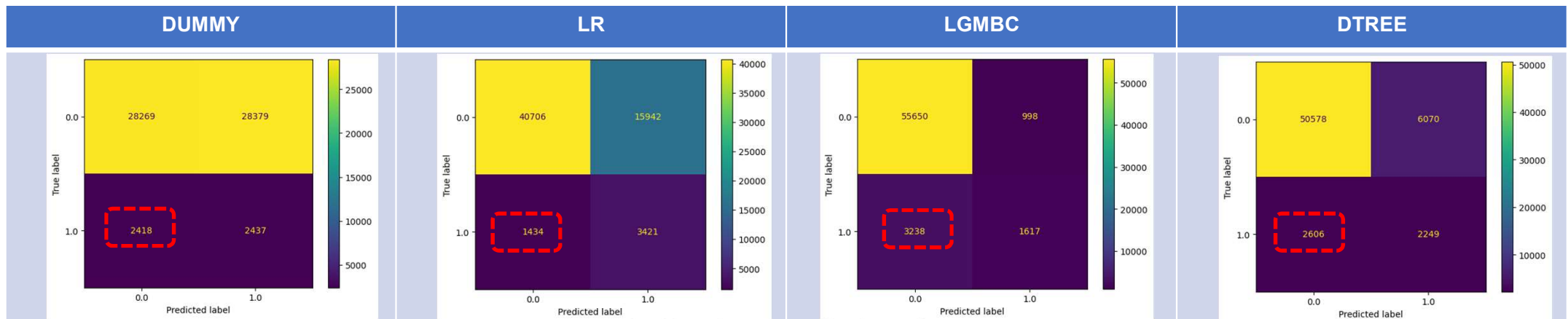


durée  
Comparing first 4 runs



Faux positif  
→ True = 1 (non solvable)  
→ prédit = 0 (solvable)

#### Matrices de confusions



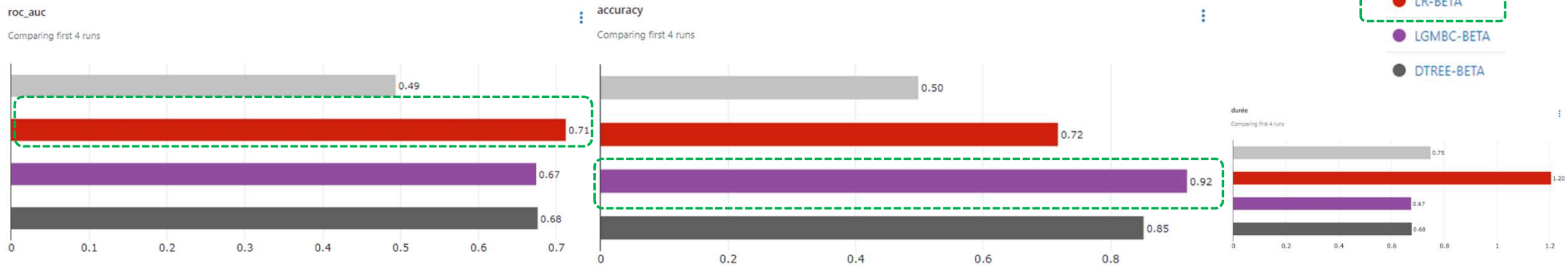
2024 Armand FAUGERE

Implémentez un modèle de scoring

### 3) Modélisations

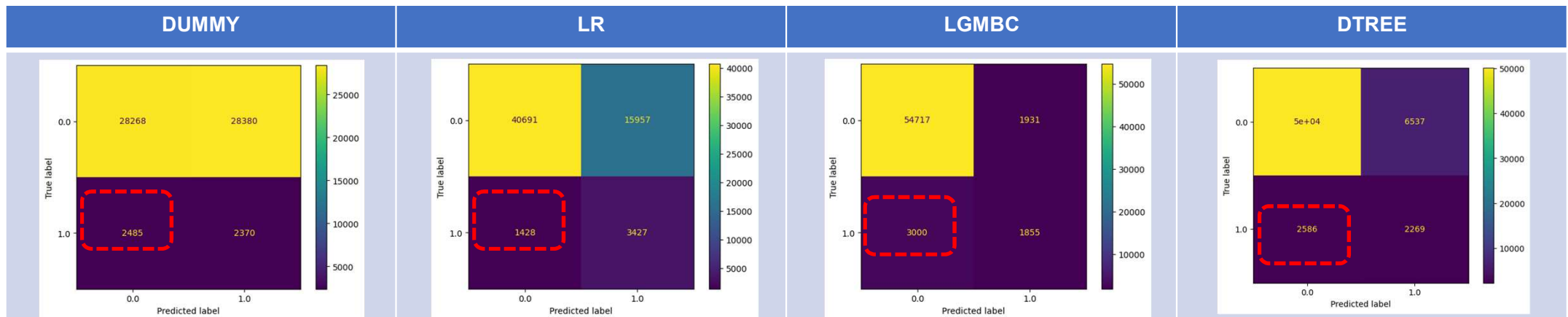


Expérimentations/Tests Optimisation métier → **betascore** ( $\beta = 2$ )



Faux positif  
→ True = 1 (non solvable)  
→ prédit = 0 (solvable)

#### Matrices de confusions



2024 Armand FAUGERE

Implémentez un modèle de scoring

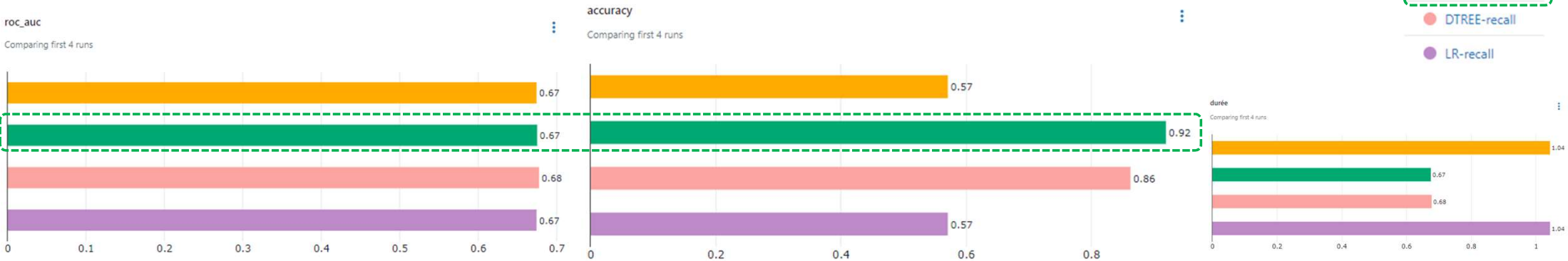
12

# 3) Modélisations



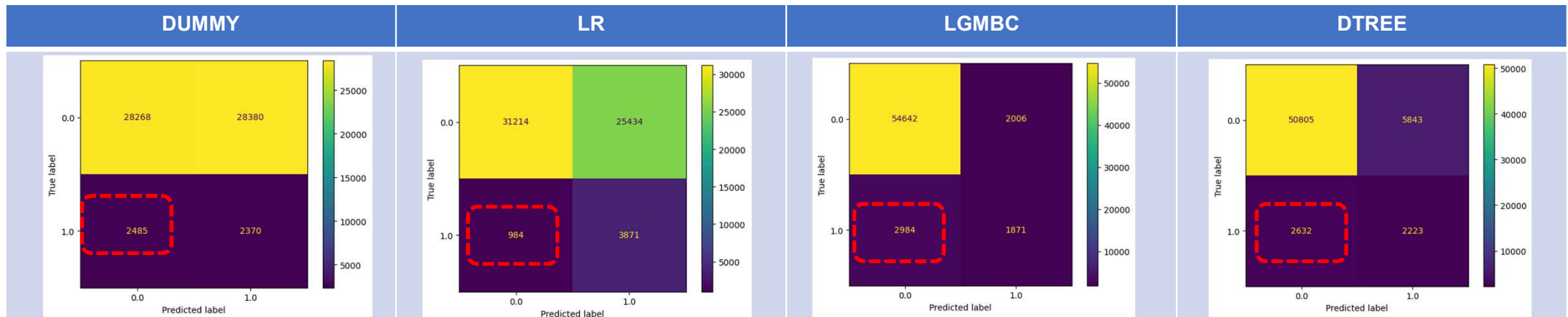
Expérimentations/Tests Optimisation métier → recall

● Dummy-recall  
● LGMBC-recall  
● DTREE-recall  
● LR-recall



Faux positif  
→ True = 1 (non solvable)  
→ prédit = 0 (solvable)

## Matrices de confusions





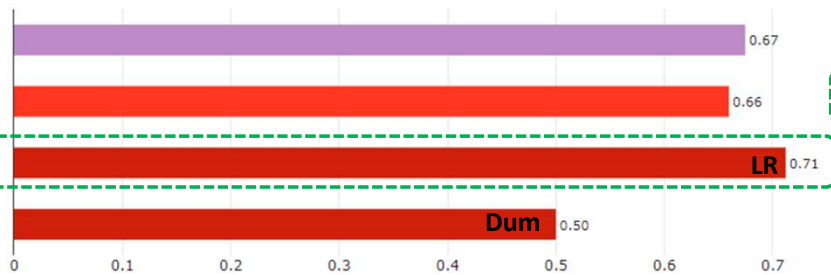
# 3) Modélisations



Expérimentations/Tests Optimisation métier → **spécificité**

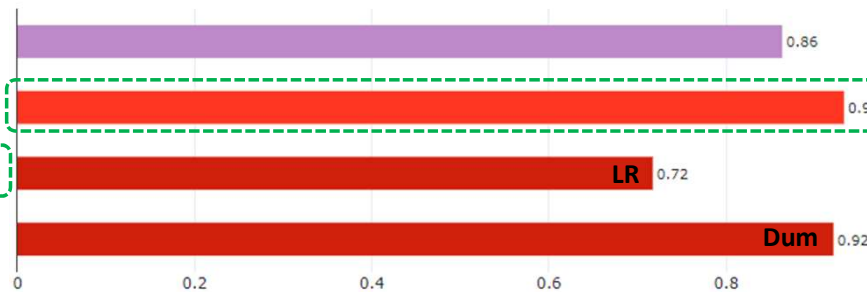
roc\_auc

Comparing first 4 runs



accuracy

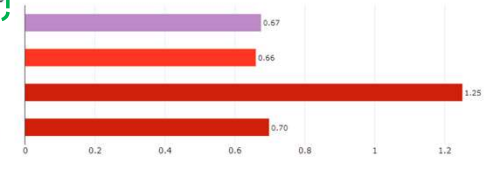
Comparing first 4 runs



- DTREE-spécificité
- LGMBC-spécificité
- LR-spécificité
- Dummy-spécificité

durée

Comparing first 4 runs



- Faux positif
- True = 1 (non solvable)
- prédit = 0 (solvable)

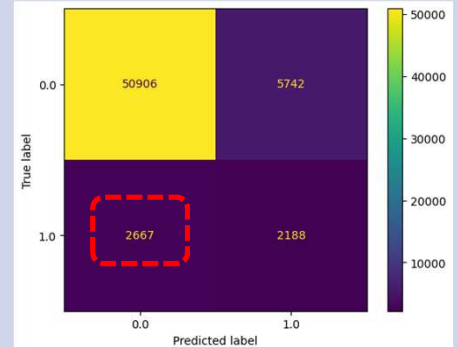
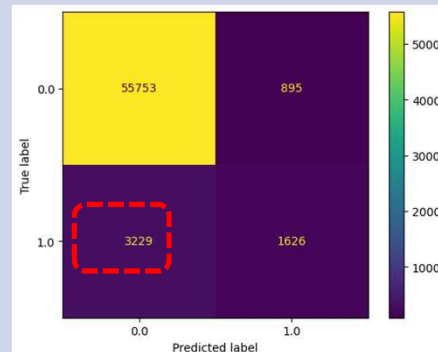
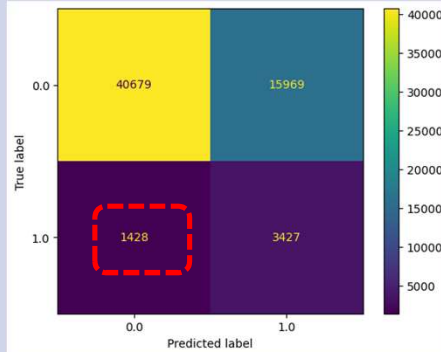
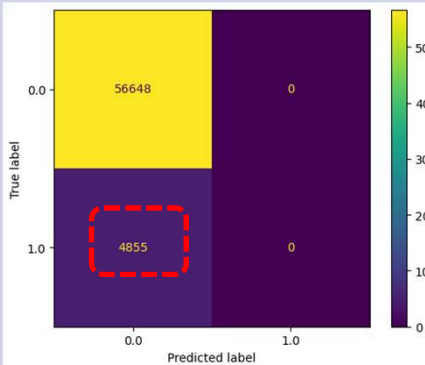
Matrices de confusions

DUMMY

LR

LGMBC

DTREE



2024 Armand FAUGERE

Implémentez un modèle de scoring

14

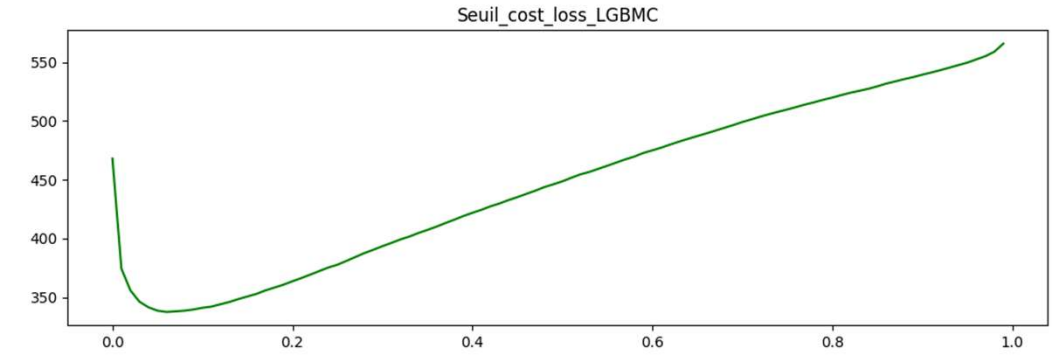
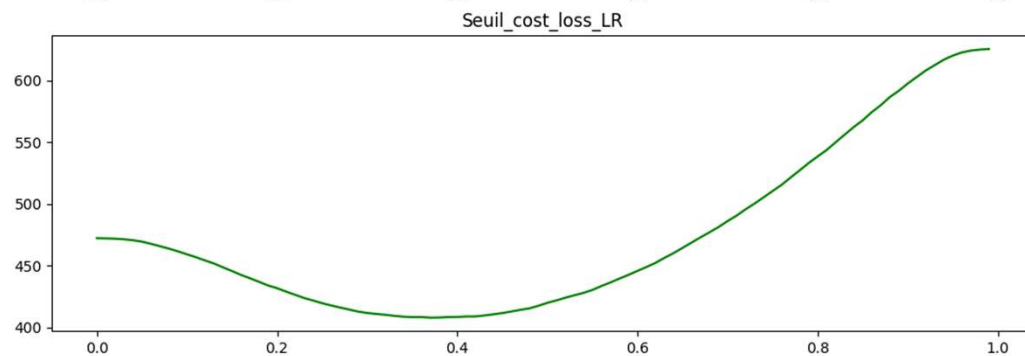
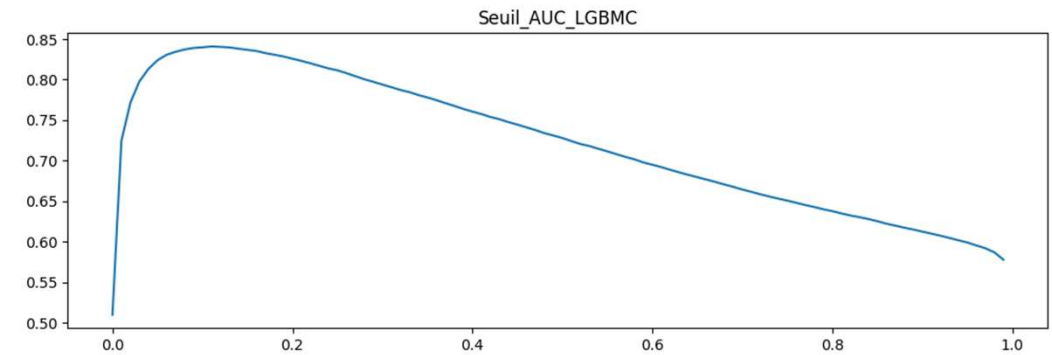
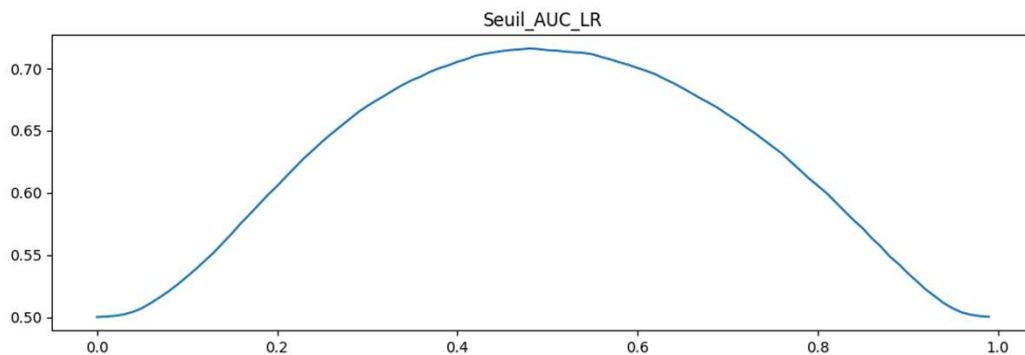
### 3) Modélisations



Expérimentations/Tests Optimisation métier → **cost\_loss + seuil optimisé**

Optimisation de seuil  
focus Logistic Regression et LGBMC

Fonction de coût =  $((2 \times \text{FN}) + (20 \times \text{FP}) + \text{TP} + \text{TN}) / 1000$



\* Au sens de la matrice de confusion

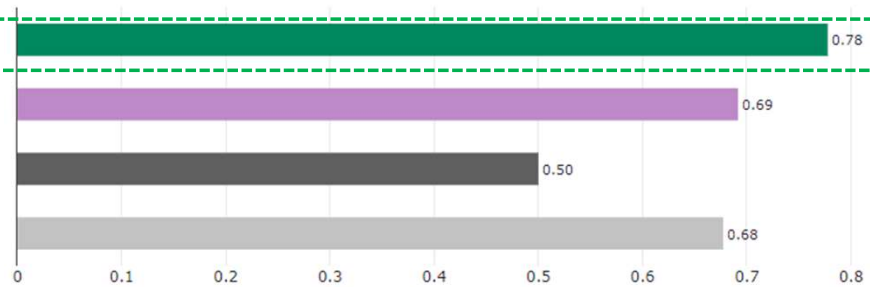
### 3) Modélisations

Expérimentations/Tests Optimisation métier → **cost\_loss + seuil optimisé**



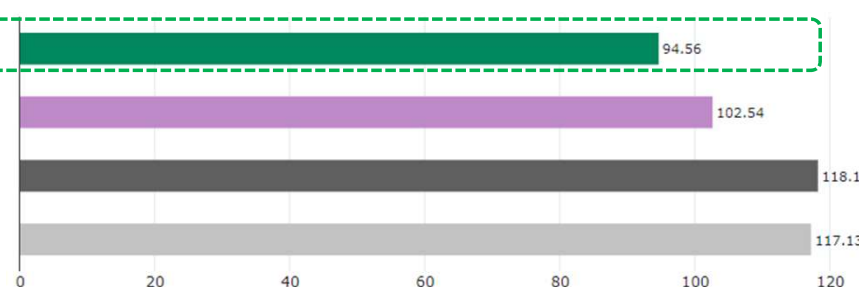
roc\_auc

Comparing first 4 runs



cost\_loss

Comparing first 4 runs



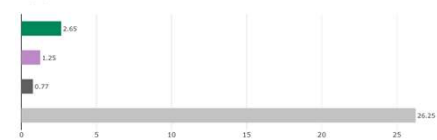
LGMBC-cost-loss

LR-cost-loss

Dum-class-cost-loss

Dtree-cost-loss

durée  
Comparing first 4 runs



Faux positif  
→ True = 1 (non solvable)  
→ prédit = 0 (solvable)

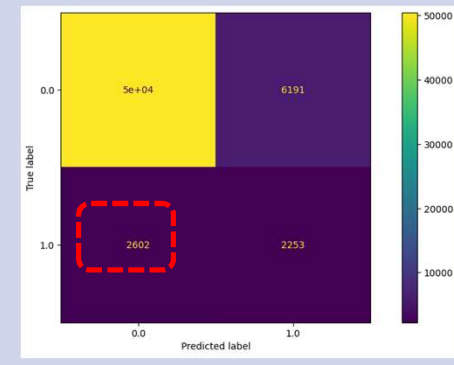
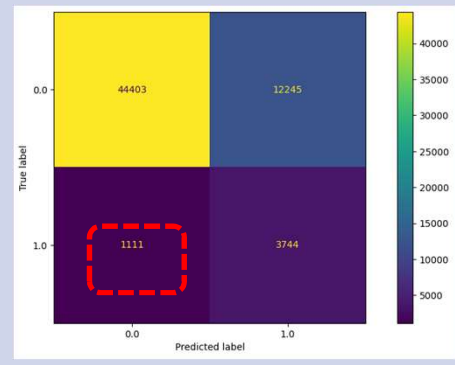
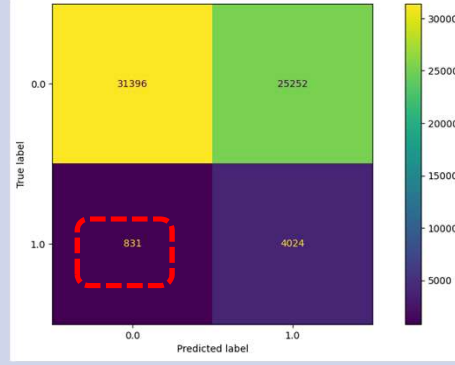
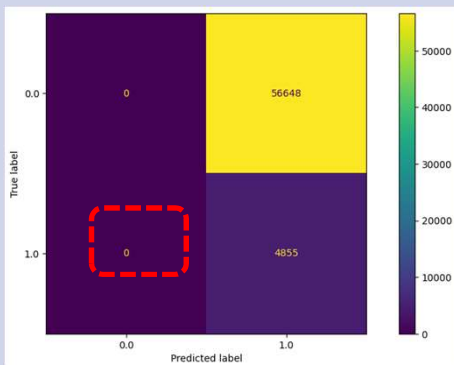
#### Matrices de confusions

DUMMY

LR

LGMBC

DTREE

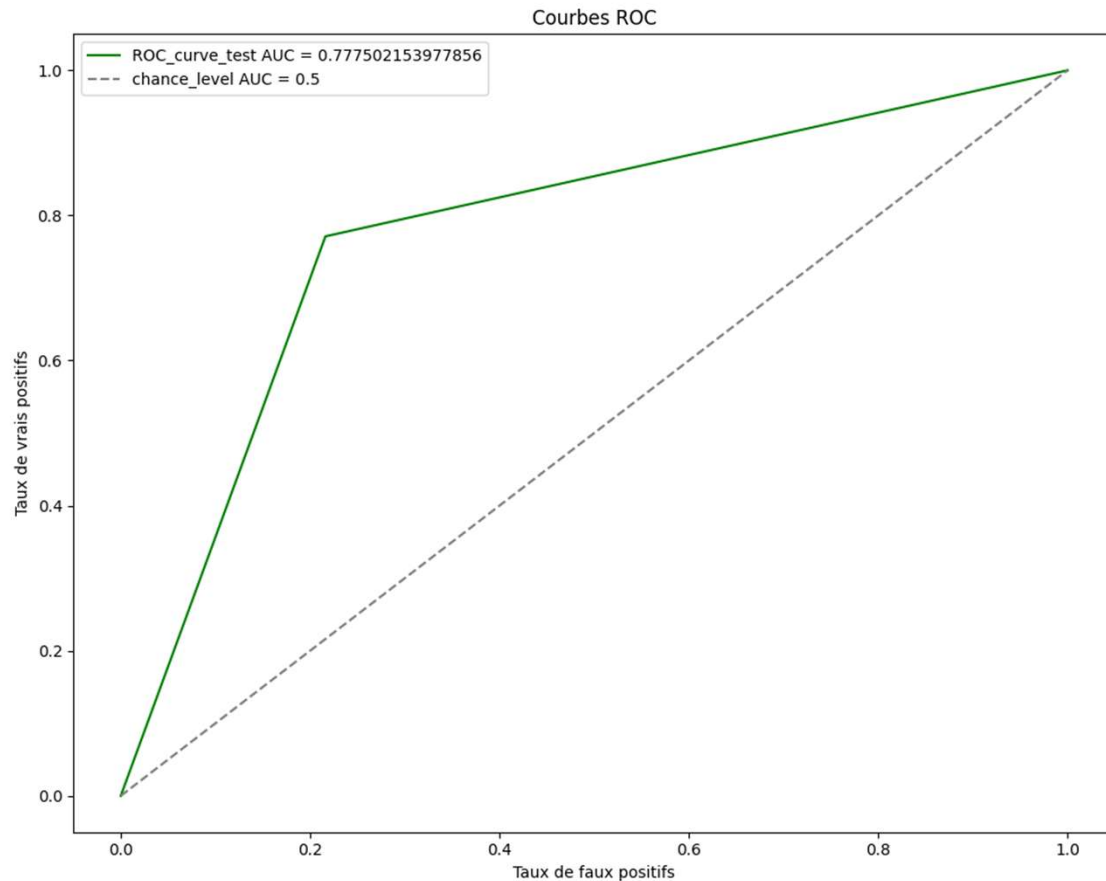


2024 Armand FAUGERE

Implémentez un modèle de scoring

16

### 3) Modélisations



#### Modèle retenu

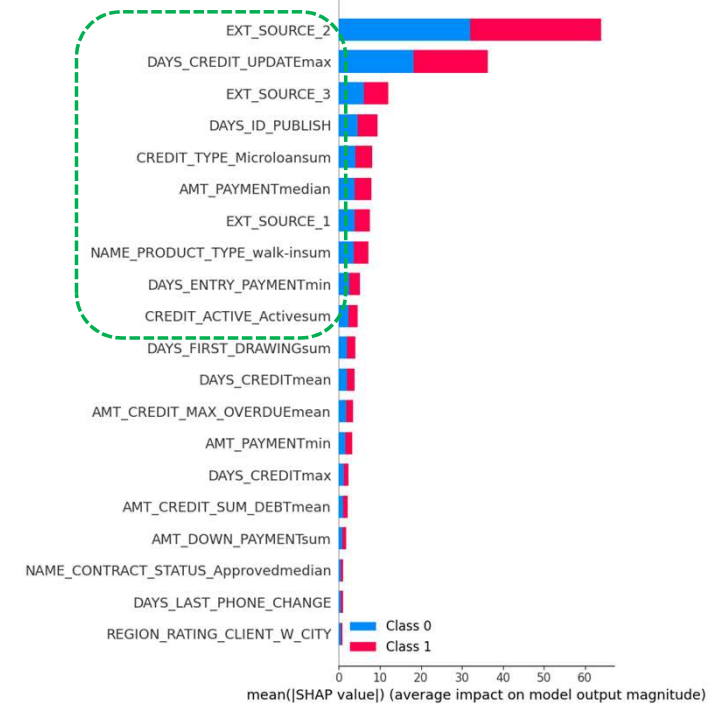
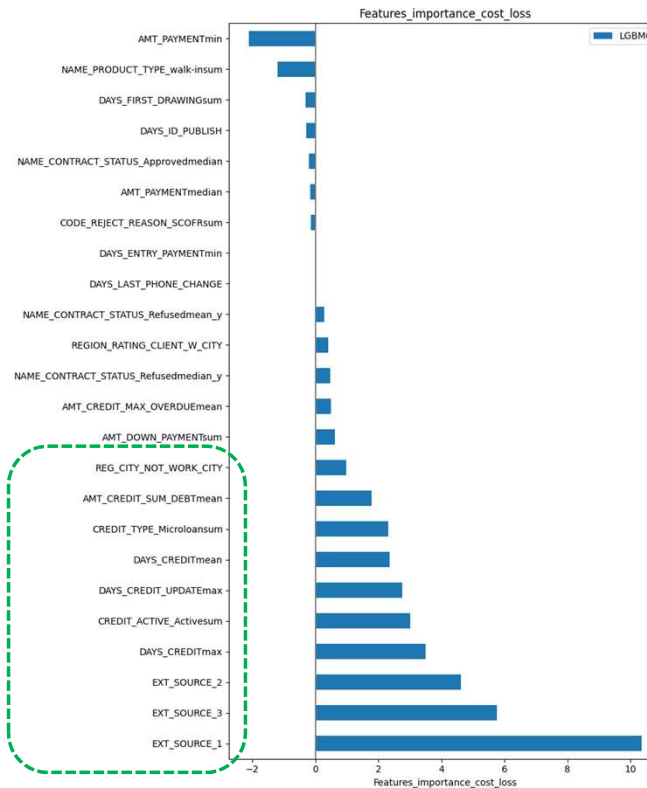
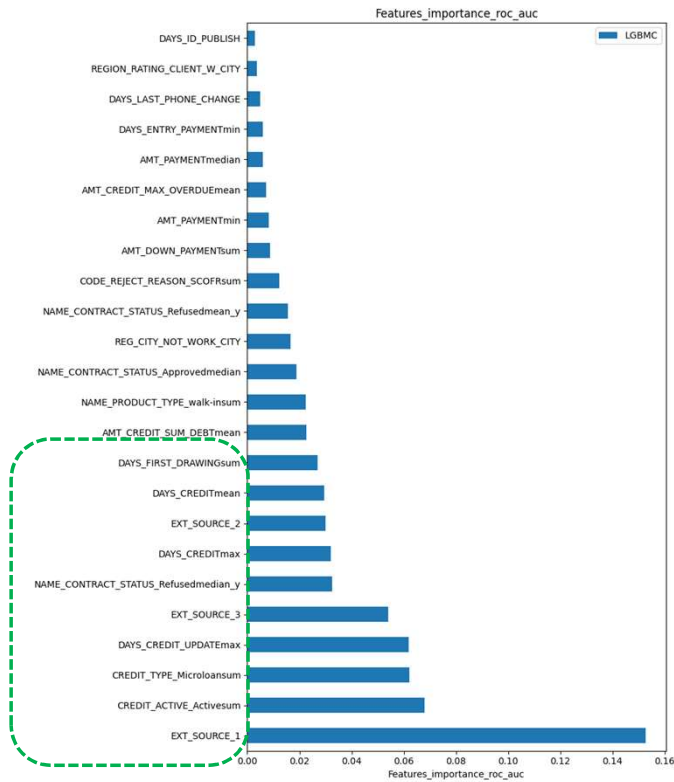
```
model_LGMBC  
{'n_estimators' : 150}  
{'random_state' : 0}  
{'learning_rate' : 0.9}  
{'seuil' : 0.07})
```

[voir mlflow tracking](#)

### 3) Modélisations



#### Features importance globale



**Permutation\_importance**  
→ Scoring = « roc\_auc »

**Permutation\_importance**  
→ Scoring = « cost\_loss »

**explainer →**  
**shap.TreeExplainer(model\_LGBMC[1])**

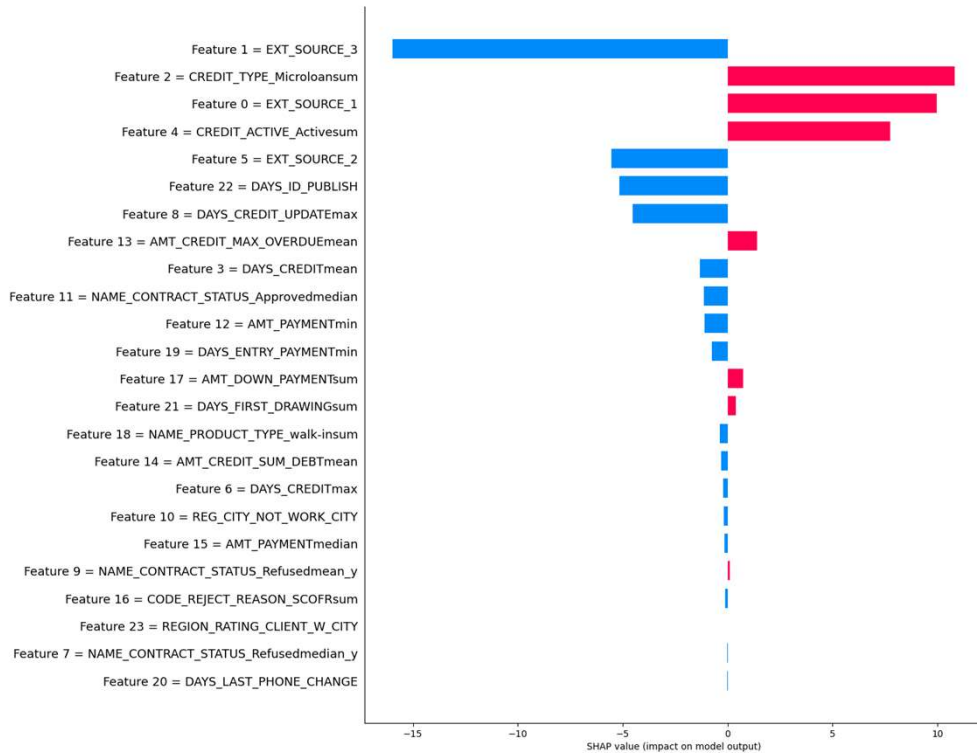


# 3) Modélisations

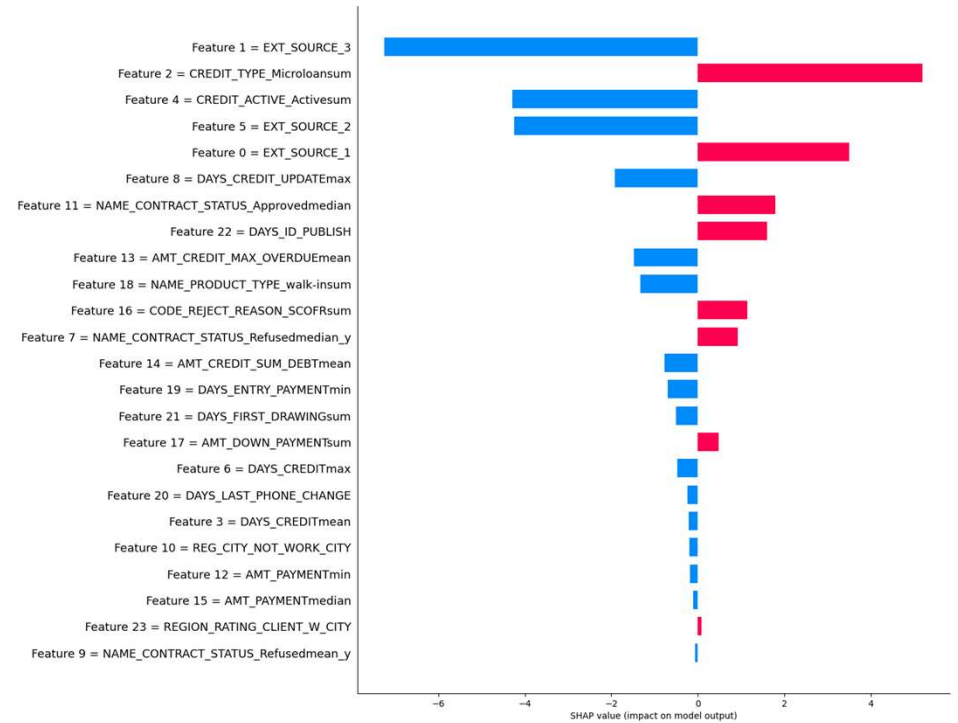


## Features importance locale → shap

**solvable**



**Non solvable**



# IV) Réalisation de l'API et déploiement

## 4) Réalisation de l'API et déploiement



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn, imblearn, time, mlflow, lightgbm, shap, joblib, fastapi, render, github

Préparation

Création et mise en œuvre de l'API en local

Déploiement en ligne

Essais en ligne

- ☐ Sérialisation du modèle  
→ model\_LGMBC\_best
- ☐ Création des données d'entrée → 200 Clients
- ☐ Création d'un répertoire de suivi et de gestion en local
- ☐ Création d'un répertoire de suivi et de gestion en ligne

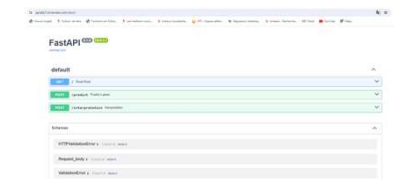
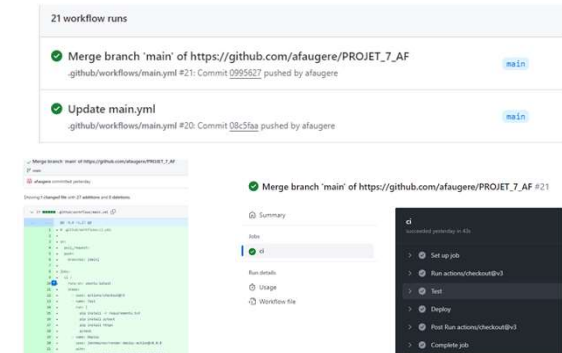
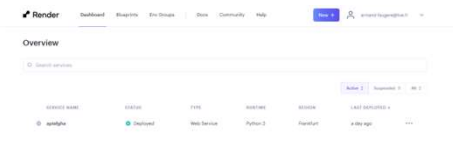
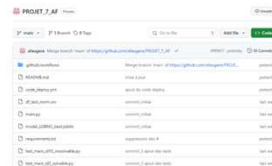
- ☐ Création de l'API avec Fastapi
- ☐ Réalisation de tests manuels
- ☐ Création et réalisation de tests en série :  
→ index3 (solvable)  
→ index10 (non solvable)

- ☐ Mise à jour du repository avec les éléments nécessaires (code, requirements, tests...)
- ☐ Création d'un compte sur render.com et connexion avec repository github
- ☐ Création d'un fichier yml dans workflows pour réaliser commit sur render
- ☐ commit et test de l'application en ligne

- ☐ Test avec nouveau commit
- ☐ Tests API en ligne

3

Mise en œuvre de l'API



## 4) Réalisation de l'API et déploiement



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn, imblearn, time, mlflow, lightgbm, shap, joblib, fastapi, render, github

### Repositories

projet\_7\_AF

.git	✓	31/01/2024 08:28	Dossier de fichiers	
.github	✓	31/01/2024 08:28	Dossier de fichiers	
.pytest_cache	✓	23/01/2024 16:42	Dossier de fichiers	
__pycache__	✓	29/01/2024 09:54	Dossier de fichiers	
archives	✓	29/01/2024 08:41	Dossier de fichiers	
.gitignore	✓	29/01/2024 08:51	Fichier source Git I...	1 Ko
code_deploy	✓	31/01/2024 08:23	Fichier source Yaml	1 Ko
df_test_norm	✓	22/01/2024 08:48	Fichier CSV Micro...	22 787 Ko
main	✓	24/01/2024 14:11	Fichier source Pyt...	2 Ko
model_LGBMC_bestjoblib	✓	18/01/2024 20:55	Fichier JOBLIB	4 249 Ko
README	✓	31/01/2024 08:21	Fichier source Mar...	2 Ko
requirements	✓	31/01/2024 08:23	Document texte	1 Ko
test_main_id3_solvable	✓	24/01/2024 14:11	Fichier source Pyt...	1 Ko
test_main_id10_nosolvable	✓	24/01/2024 14:11	Fichier source Pyt...	1 Ko

push

pull

afaugere / PROJET\_7\_AF

PROJET\_7\_AF Private

Unwatch 1

main 1 Branch 0 Tags

Go to file t Add file <> Code

afaugere Merge branch 'main' of https://github.com/afaugere/PROJET\_7\_AF ✓ 0995627 · yesterday 30 Commits

.github/workflows	Merge branch 'main' of https://github.com/afaugere/PROJE...	yesterday
README.md	mise à jour	yesterday
code_deploy.yml	ajout du code deploy	yesterday
df_test_norm.csv	commit_initial	last week
main.py	commit_initial	last week
model_LGBMC_bestjoblib	commit_initial	last week
requirements.txt	suppression des #	yesterday
test_main_id10_nosolvable.py	commit_3 ajout des tests	last week
test_main_id3_solvable.py	commit_3 ajout des tests	last week

## 4) Réalisation de l'API et déploiement



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn, imblearn, time, mlflow, lightgbm, shap, joblib, fastapi, render, github

### Extrait de code API

```
@app.get("/")
def read_root():
    return {"message": "Welcome to the AF ML Model API"}

# Définition des requêtes
class Request_body(BaseModel):
    ID_CLIENT: int

# Endpoint prediction probabilité
@app.post('/predict')
# fonction de prédiction
def predict_labels(input_data : Request_body) :
    # nouvelles données
    input_data = input_data.dict()
    # prédiction
    client = df.loc[input_data["ID_CLIENT"]].values
    client = np.reshape(client, (1,-1))
    prediction_proba = model.predict_proba(client)
    # définition du seuil
    seuil = 0.07
    # colonne avec prediction 1
    prediction_proba_one = prediction_proba[:,1]
    # colonne avec seuil
    predict_proba_label = (prediction_proba_one > seuil).astype(int)
    # return f" label : {predict_proba_label}, probability : {prediction_proba_one}"
    return {'label' : predict_proba_label.tolist(), 'probability' : prediction_proba_one.tolist()}

# Endpoint interprétabilité
@app.post('/interpretation')
# Fonction interprétation locale
def interpretation(input_data : Request_body) :
    input_data = input_data.dict()
    # Interprétation
    client = df.loc[input_data["ID_CLIENT"]].values
    client = np.reshape(client, (1,-1))
    explainer = shap.TreeExplainer(model[1])
    shap_values = explainer.shap_values(client)
    df_shap_values = pd.DataFrame(shap_values[1], columns = df.columns)
    return df_shap_values
```

Endpoint probabilité :

- Prend en données d'entrée l'index (request\_body)
- ➔ Retourne la classification
- ➔ Retourne la probabilité

Endpoint interprétation :

- Prend en données d'entrée l'index (request\_body)
- ➔ Dataframe avec features importances localement



## 4) Réalisation de l'API et déploiement



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn, imblearn, time, mlflow, lightgbm, shap, joblib, fastapi, render, github

- ☐ Test de bonne connexion de l'application
- ☐ Test retour sur résultat attendu :
  - Index 3 → solvable
  - Index 10 → non solvable

Essais en local

Essais lors du déploiement

```
PS C:\Users\ARMAN\OneDrive\Bureau\DATASCIENCE\9-OPEN CLASSROOMS\Projet 7-Implémentez un modèle de scoring\projet_7_AF> pytest
===== test session starts =====
platform win32 -- Python 3.8.18, pytest-7.4.4, pluggy-1.3.0
rootdir: C:\Users\ARMAN\OneDrive\Bureau\DATASCIENCE\9-OPEN CLASSROOMS\Projet 7-Implémentez un modèle de scoring\projet_7_AF
plugins: anyio-3.5.0, typeguard-2.13.3
collected 4 items

test_main_id10_nosolvable.py ..
test_main_id3_solvable.py ..

===== warnings summary =====
test_main_id10_nosolvable.py::test_classifier
test_main_id3_solvable.py::test_classifier
  Usage of np.ndarray subset (sliced data) is not recommended due to it will double the peak memory cost in LightGBM.

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
===== 4 passed, 2 warnings in 3.26s =====
PS C:\Users\ARMAN\OneDrive\Bureau\DATASCIENCE\9-OPEN CLASSROOMS\Projet 7-Implémentez un modèle de scoring\projet_7_AF>
```

✓ Merge branch 'main' of https://github.com/afaugere/PROJET\_7\_AF #21

Summary

Jobs

✓ ci

Run details

Usage

Workflow file

ci

succeeded yesterday in 43s

- > ✓ Set up job
- > ✓ Run actions/checkout@v3
- > ✓ Test
- > ✓ Deploy
- > ✓ Post Run actions/checkout@v3
- > ✓ Complete job

## 4) Réalisation de l'API et déploiement



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn, imblearn, time, mlflow, lightgbm, shap, joblib, fastapi, render, github

### Déploiement en ligne

21 workflow runs

- ✓ Merge branch 'main' of https://github.com/afaugere/PROJET\_7\_AF  
github/workflows/main.yml #21: Commit 0995627 pushed by afaugere
- ✓ Update main.yml  
github/workflows/main.yml #20: Commit 08c5faa pushed by afaugere

### Code fichier .yaml pour déploiement

✓ Merge branch 'main' of https://github.com/afaugere/PROJET\_7\_AF

main

afaugere committed yesterday

Showing 1 changed file with 27 additions and 0 deletions.

```
... 27 github/workflows/main.yml
... @@ -0,0 +1,27 @@
1 + # github/workflows/ci.yml
2 +
3 + on:
4 +   pull_request:
5 +   push:
6 +     branches: [main]
7 +
8 + jobs:
9 +   ci:
10 +     runs-on: ubuntu-latest
11 +     steps:
12 +       - uses: actions/checkout@v3
13 +       - name: Test
14 +         run: |
15 +           pip install -r requirements.txt
16 +           pip install pytest
17 +           pip install httpx
18 +           pytest
19 +       - name: Deploy
20 +         uses: johnbeynon/render-deploy-action@v0.0.8
21 +         with:
22 +           service-id: ${ secrets.RENDER_SERVICE_ID }
23 +           api-key: ${ secrets.RENDER_DEPLOY_HOOK_URL }
24 +           wait-for-success: true
25 +
```

2024 Armand FAUGERE

### Synthèse déploiement

✓ Merge branch 'main' of https://github.com/afaugere/PROJET\_7\_AF #21

Summary

Jobs

✓ ci

Run details

Usage

Workflow file

ci  
succeeded yesterday in 43s

- > ✓ Set up job
- > ✓ Run actions/checkout@v3
- > ✓ Test
- > ✓ Deploy
- > ✓ Post Run actions/checkout@v3
- > ✓ Complete job

Implémentez un modèle de scoring

### Interface API

apiafp7.onrender.com/docs

Nouvel onglet Python, de zéro Fonctions en Python... Les meilleurs cours...

FastAPI 0.1.0 OAS 3.1  
/openapi.json

### default

GET / Read Root

POST /predict Predict Labels

POST /interpretation Interpretation

### Schemas

HTTPValidationError > Expand all object

Request\_body > Expand all object

ValidationError > Expand all object

[voir API](#)

25

# V) Analyse du data drift

## 5) Analyse du data drift



Jupyter Notebook, Python, Pandas, Numpy, Matplotlib, Seaborn, sklearn, imblearn, time, mlflow, lightgbm, evidently

Réalisation des  
prédictions sur jeu de  
test

Analyse data drift

**Pas de datadrift**  
Drift détecté pour 12% des  
features (3/25)

4

**Suivi  
fonctionnement  
API**

**DATA DRIFT**

- ☐ Utilisation du modèle  
→ `model_LGMBC_best`  
sur jeu de TEST
- ☐ Ajout des résultats de  
prédictions au dataframe  
de TEST avec les autres  
features

- ☐ Utilisation de l'outil analyse de  
datadrift d'evidently
- set de TRAIN et target
- set de TEST et prédictions
- ☐ Analyse de l'impact sur les  
distributions de chacune des  
features



## 5) Analyse du data drift

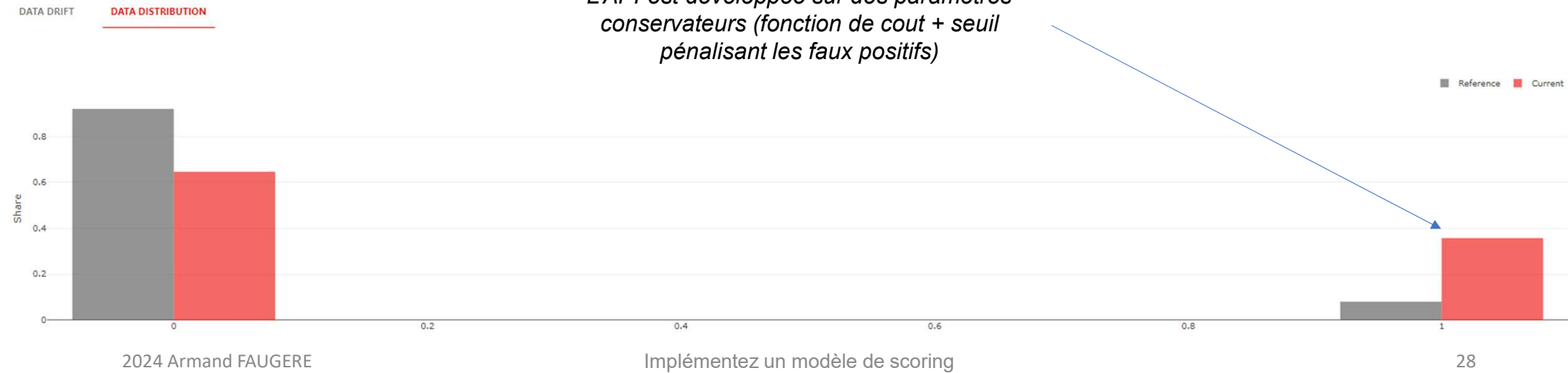


### Features avec datadrift

Feature	Type	Reference Distribution	Current Distribution	Data Drift	Stat Test	Drift Score
> TARGET	num			Detected	Jensen-Shannon distance	0.24305
> DAYS_LAST_PHONE_CHANGE	num			Detected	Wasserstein distance (normed)	0.138978
> CREDIT_TYPE_Microloansum	num			Detected	Wasserstein distance (normed)	0.103243

### TARGET

*L'API est développée sur des paramètres conservateurs (fonction de coût + seuil pénalisant les faux positifs)*











# VI) Conclusion

## 6) Conclusion



Développer et mettre en production un modèle de scoring pour prédire la probabilité de faillite d'un client, et sa classification en « refusé » ou « accepté », avec une démarche MLOPS		
Réaliser une analyse exploratoire et sélectionner les variables pertinentes	Analyse réalisée avec le choix de 24 variables : <ul style="list-style-type: none"> <li>Analyse/préparation des données (fusions, agrégations...)</li> <li>Sélection itérative des variables (corrélations, taux de remplissage, multi colinéarité)</li> <li>Analyse univariée et bi-variée</li> <li>Validation des variables avec feature_importance</li> </ul>	
Modéliser, optimiser et choisir le meilleur modèle	Choix du meilleur modèle LGMBC avec critères de fonction de coût et auc_score : <ul style="list-style-type: none"> <li>Optimisation paramétrique et métier réalisée sur 4 algorithmes différents</li> <li>Mise en place d'un Tracking des expériences via ML FLOW (20 modélisations différentes)</li> </ul>	
Analyser l'impact des variables retenues	Analyse d'impact des features réalisée au global et en local : <ul style="list-style-type: none"> <li>Au global avec feature_importance (auc_score &amp; cost_loss) et shap</li> <li>En local avec shap sur client solvable et non solvable</li> </ul>	
Réaliser une API et la déployer avec les tests associés	API réalisée et testée en local, puis déployée via GitHub actions sur render.com avec les tests automatisés (condition pour déployer l'API)	
Analyser en production le data drift	Le datadrift a été analysée sur la base du jeu de test via evidently : <ul style="list-style-type: none"> <li>Pas de datadrift constaté (3/25 des features concernées)</li> </ul>	

# Merci

- Armand FAUGERE
- [armand-faugere@live.fr](mailto:armand-faugere@live.fr)

