

# Prêt à dépenser



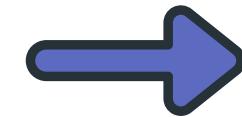
Implémenter un outil de scoring

# Objectifs du projet

Prêt à  
dépenser

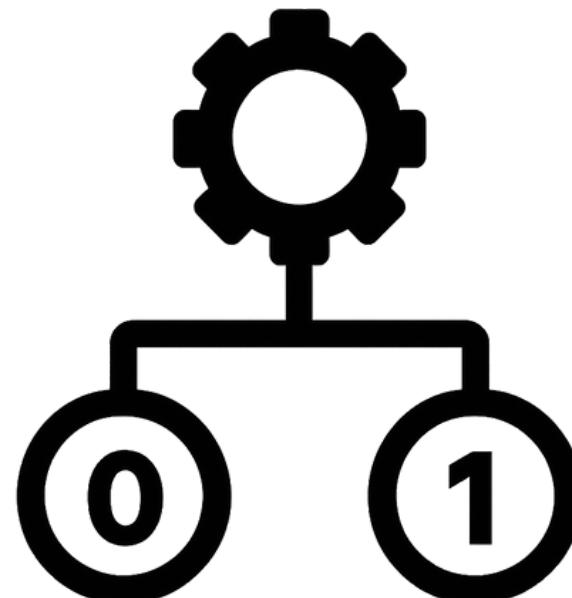


Gestion du  
risque



Outil de  
scoring

Implémenter un outil de  
scoring



Suivre l'optimisation via  
MLFlow

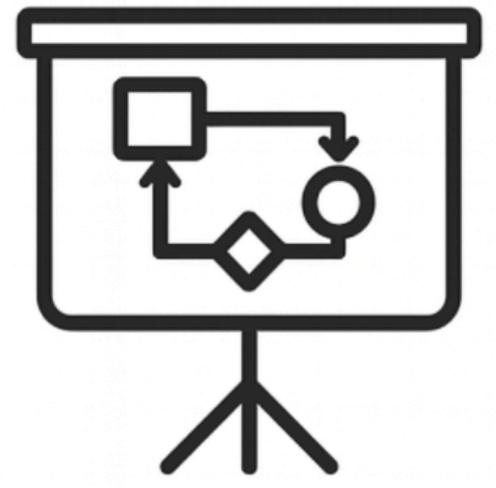


Deployer le modèle dans  
le cloud





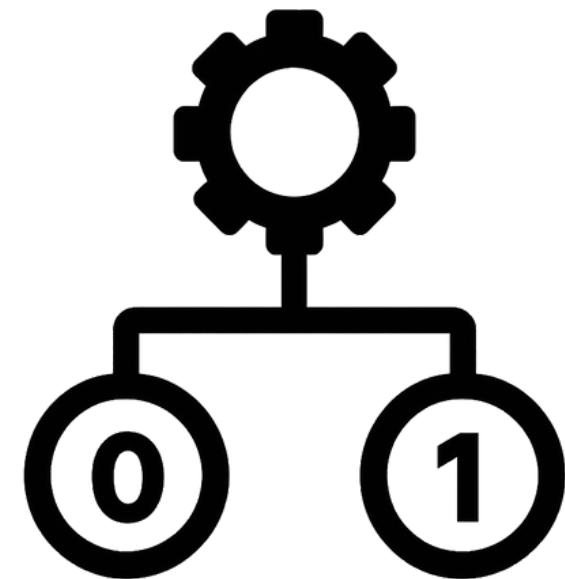
Présentation  
des données



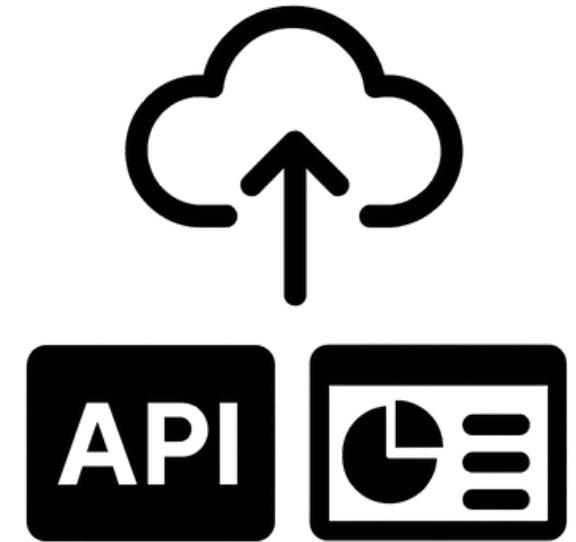
Présentation  
de l'approche



EDA + feature  
engineering



Modélisation



API +  
Dashboard

# Présentation des données

# Présentation de l'approche

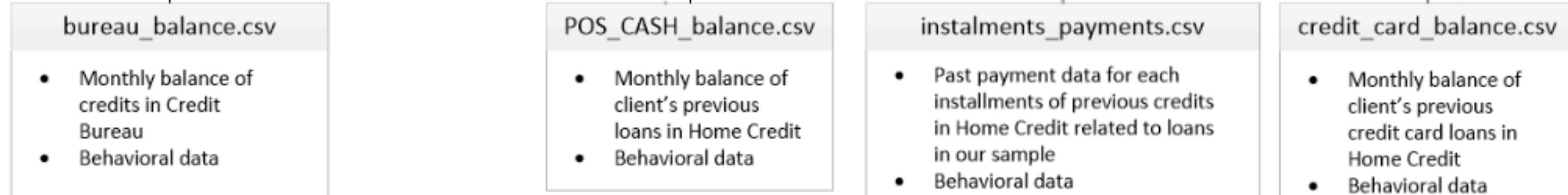
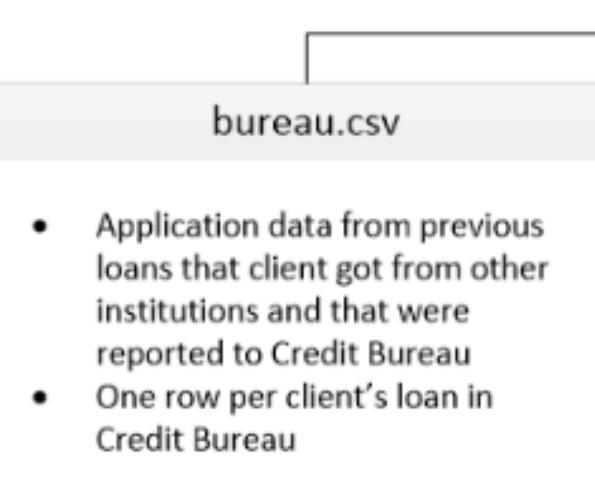
# EDA + feature engineering

# Modélisation

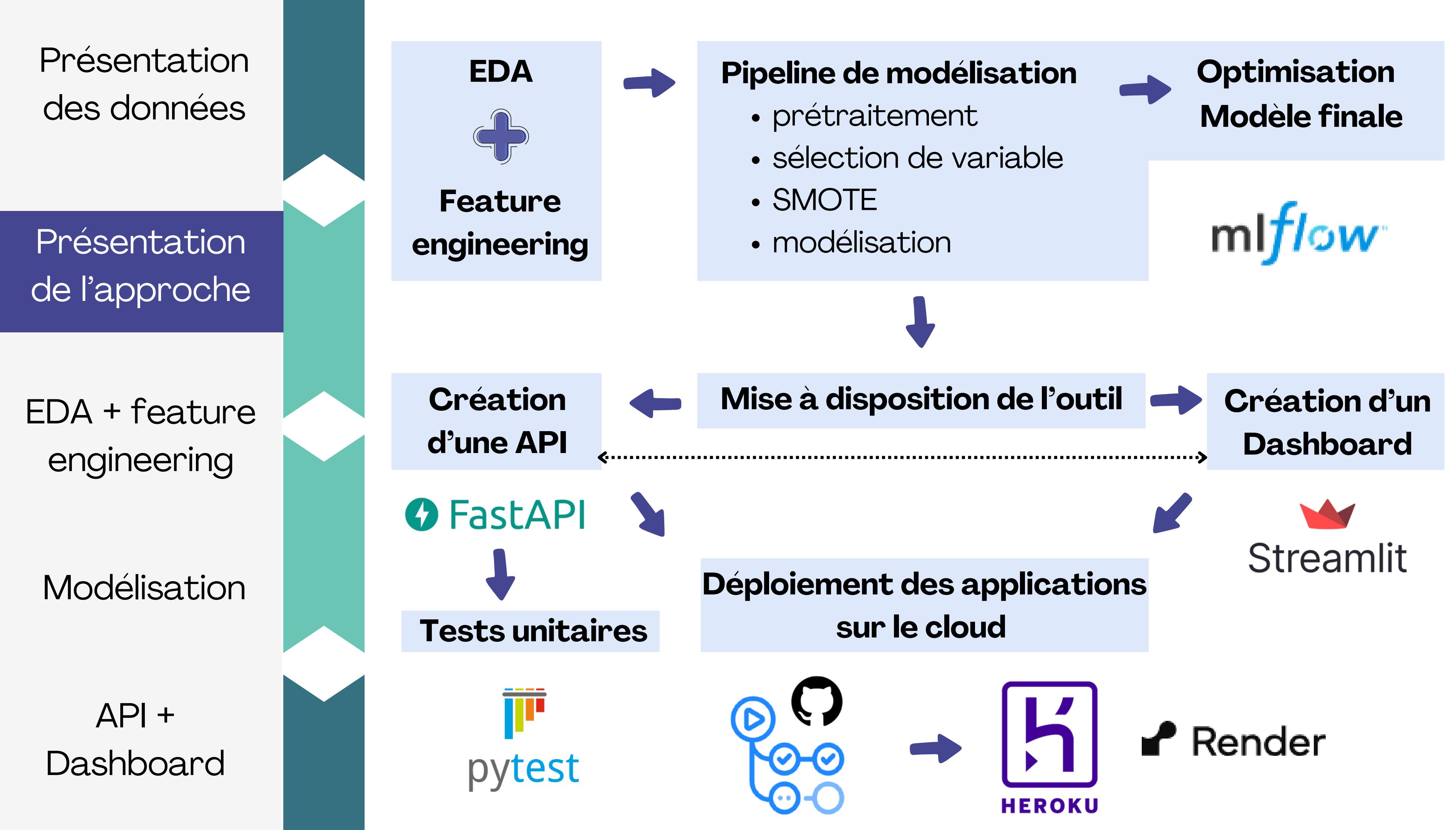
# API + Dashboard

## Table des demandes

### Précédent crédits externes



### Précédent crédits internes



# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard

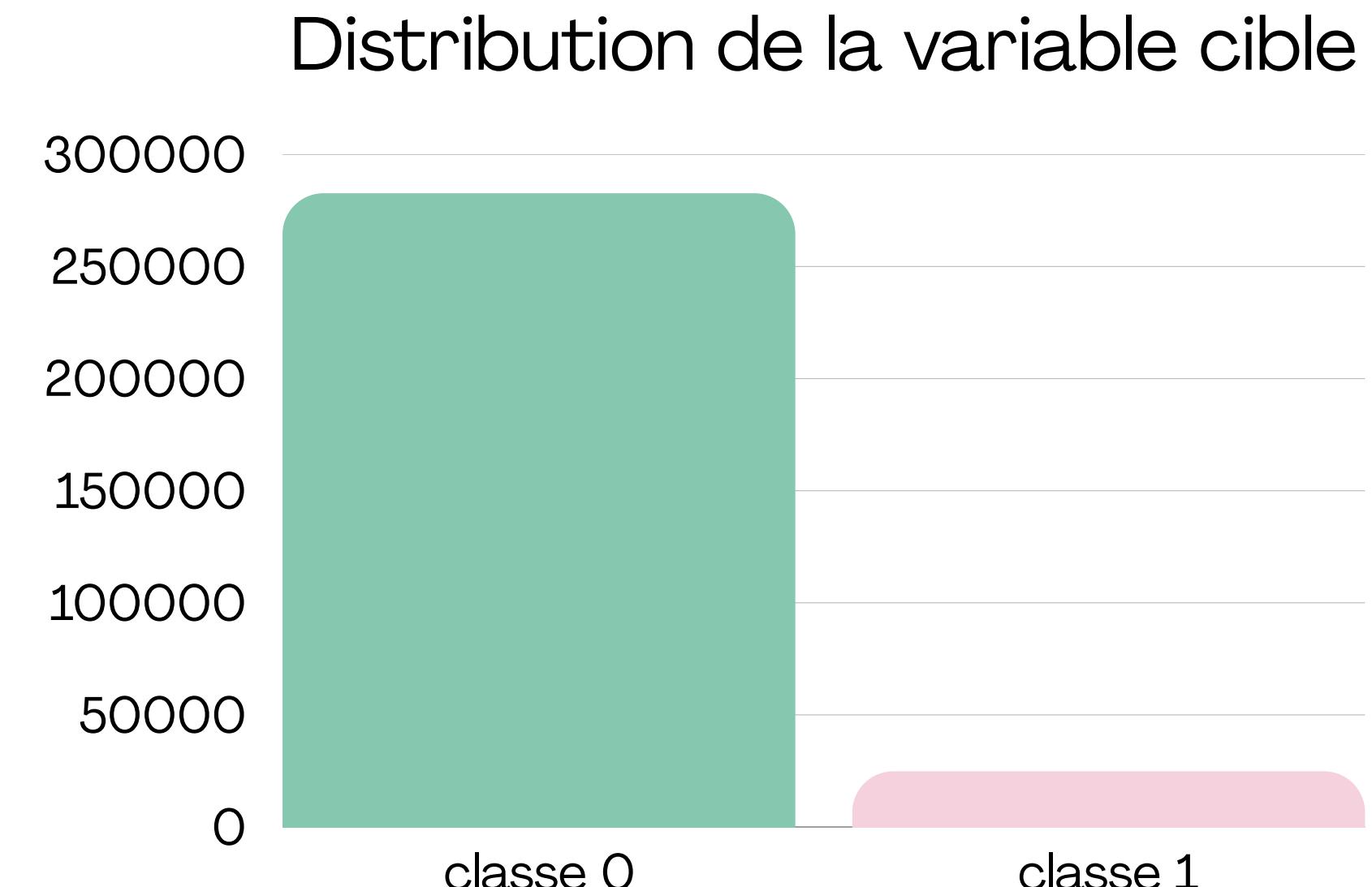
# Séparation des données

# Taux de valeurs manquantes

# Nettoyage

# Feature engineering

# Aggrégation



# Jeux de données fortement déséquilibré

## Stratification

Présentation  
des données

Présentation  
de l'approche

EDA + feature  
engineering

Modélisation

API +  
Dashboard

Séparation des  
données

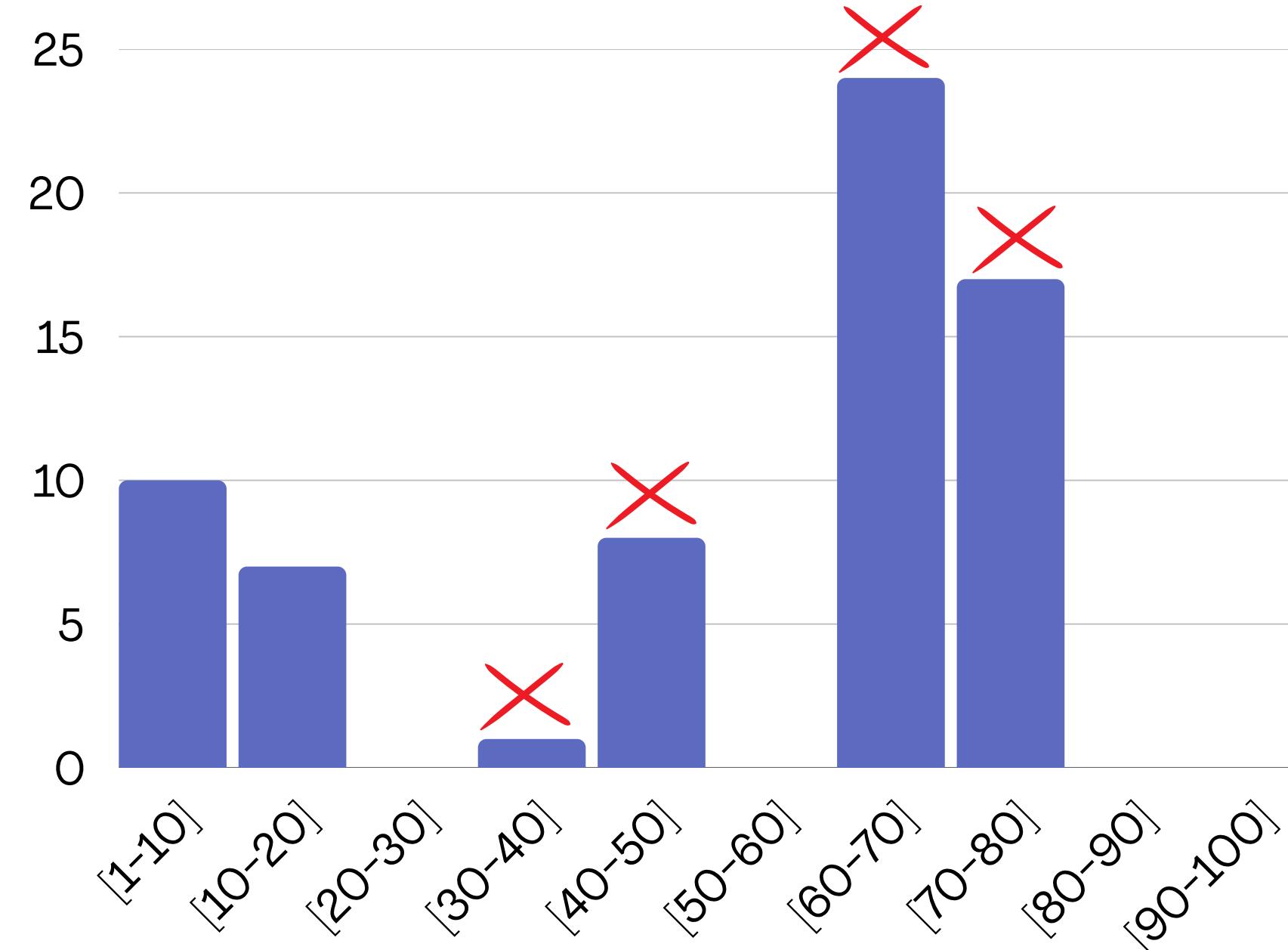
Taux de valeurs  
manquantes

Nettoyage

Feature  
engineering

Aggrégation

Distribution du taux de valeurs  
manquante dans la table principale



121 variables

67 variables avec valeurs manquantes

50 variables supprimées

Présentation  
des données

Présentation  
de l'approche

EDA + feature  
engineering

Modélisation

API +  
Dashboard

## Séparation des données

## Taux de valeurs manquantes

## Nettoyage

## Feature engineering

## Aggrégation

Salaire > 1 000 000

Nombre de jours employé = 45632

Variables temporelles < 0

Nombre de jours depuis la naissance

Variables avec plus de 10 modalités

Valeur  
manquante



valeur absolue

Années

X

ou regroupement  
de modalité

Présentation  
des données

Présentation  
de l'approche

EDA + feature  
engineering

Modélisation

API +  
Dashboard

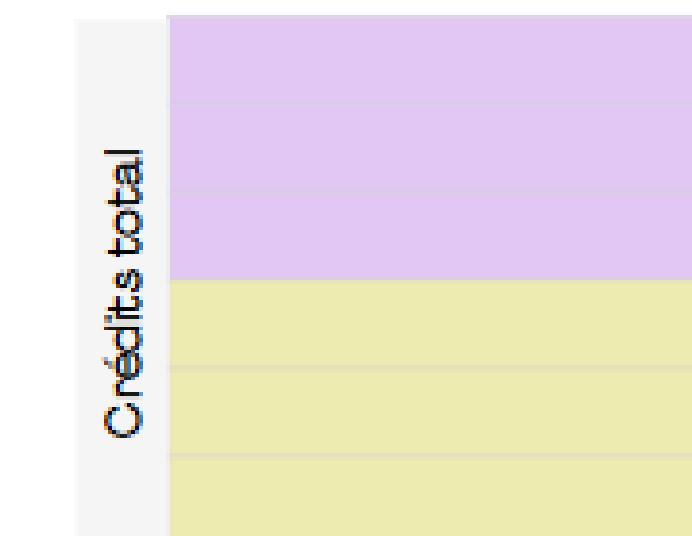
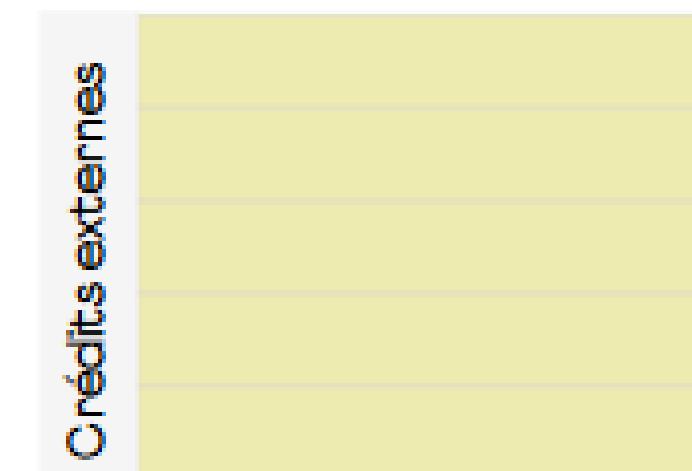
## Séparation des données

## Taux de valeurs manquantes

## Nettoyage

## Feature engineering

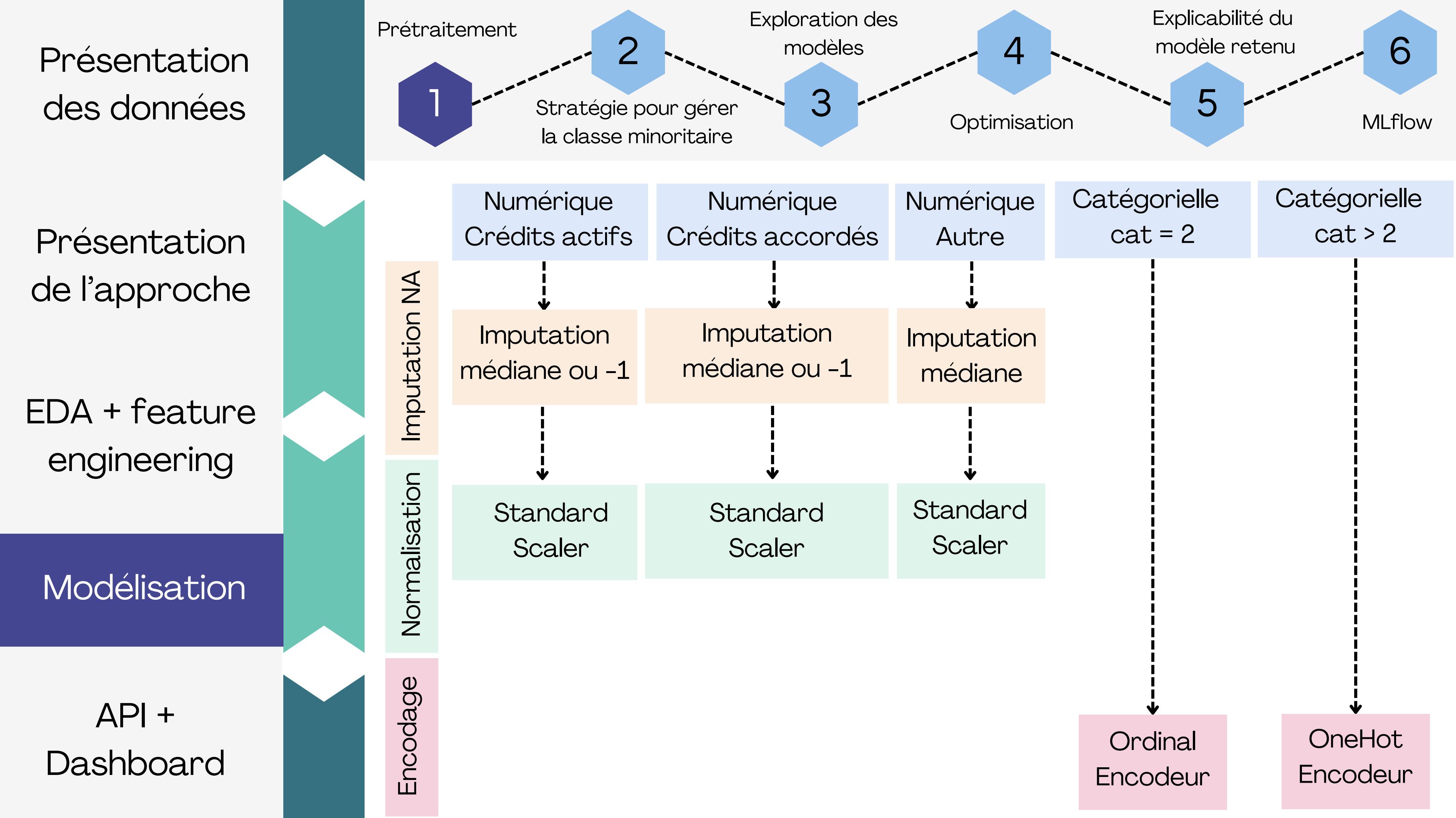
## Aggrégation



Montant mensuel à rembourser
Montant total du crédit
Crédit approuvés interne
Crédit demandés interne
Crédits actifs
Type de crédits
Durée total crédit
Dette restante
Nombre jour restant
Prolongation crédit
Nombre de jour de retard
Nombre d'échéance en retard
Nombre d'échéance total

## Aggrégation

27 nouvelles variables  
liées aux précédents  
crédits accordés et  
aux crédits actifs



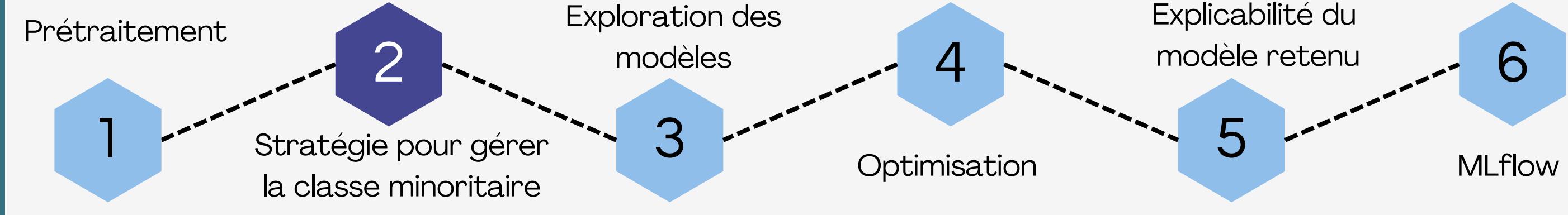
## Présentation des données

## Présentation de l'approche

## EDA + feature engineering

## Modélisation

## API + Dashboard



## SMOTE

Original Data

Resampled Data

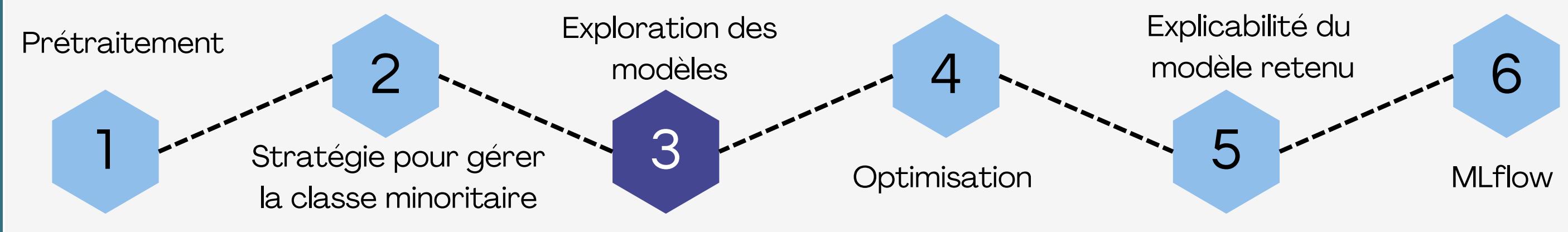
Ryan Roepke

<https://dataknowsall.com/blog/imbalanced>

Random undersampling

Random oversampling

# Présentation des données



# Présentation de l'approche

## Pipeline

Prétraitement  
SMOTE  
Modèle

## Modèles testés

LogisticRegression  
Random Forest  
LighGBM

## Evaluation

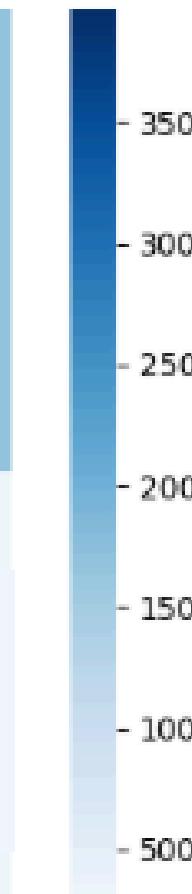
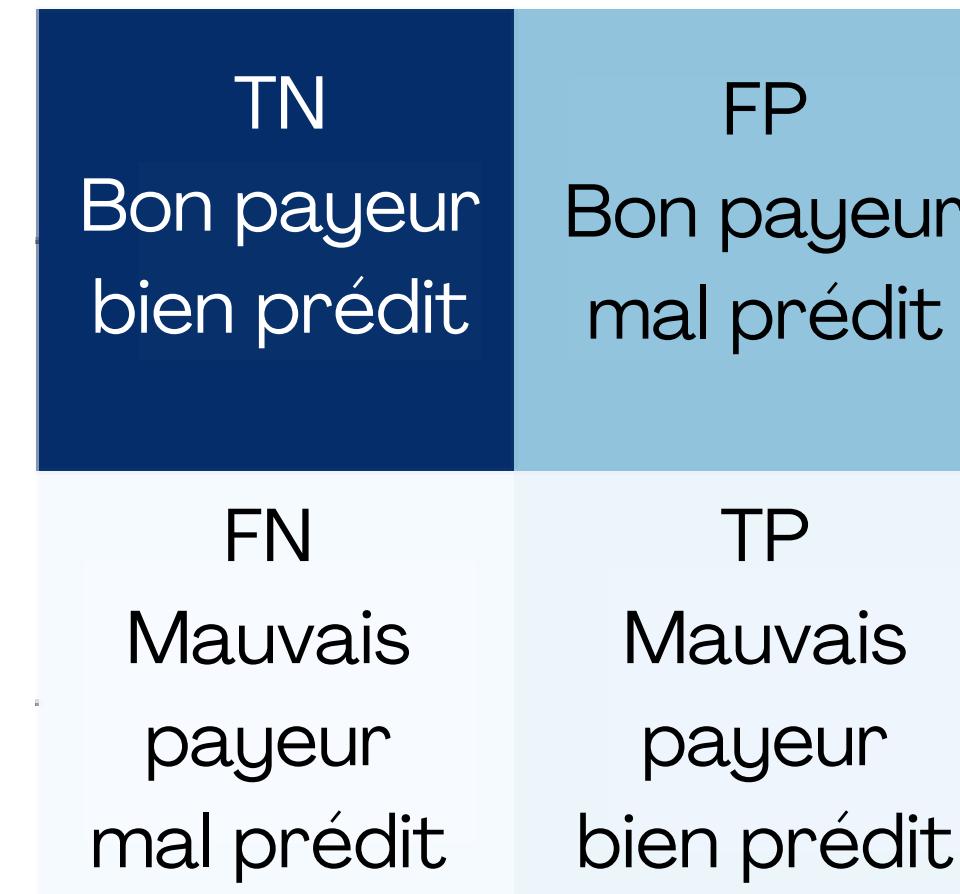
Matrice confusion  
Score de rappel  
Score F2

Validation croisée sur fold stratifiés

# EDA + feature engineering

# Modélisation

# API + Dashboard



## Score métier

Prêt: 10 000 - intérêt: 1 000

TN: + 1 000

FP : - 1 000

FN : -1 000

TP: + 10 000

$$\text{Score} = \frac{10 * \text{TP} + \text{TN} - 10 \text{FN} - \text{FP}}{\text{TP} + \text{TN} + \text{FN} + \text{FP}}$$

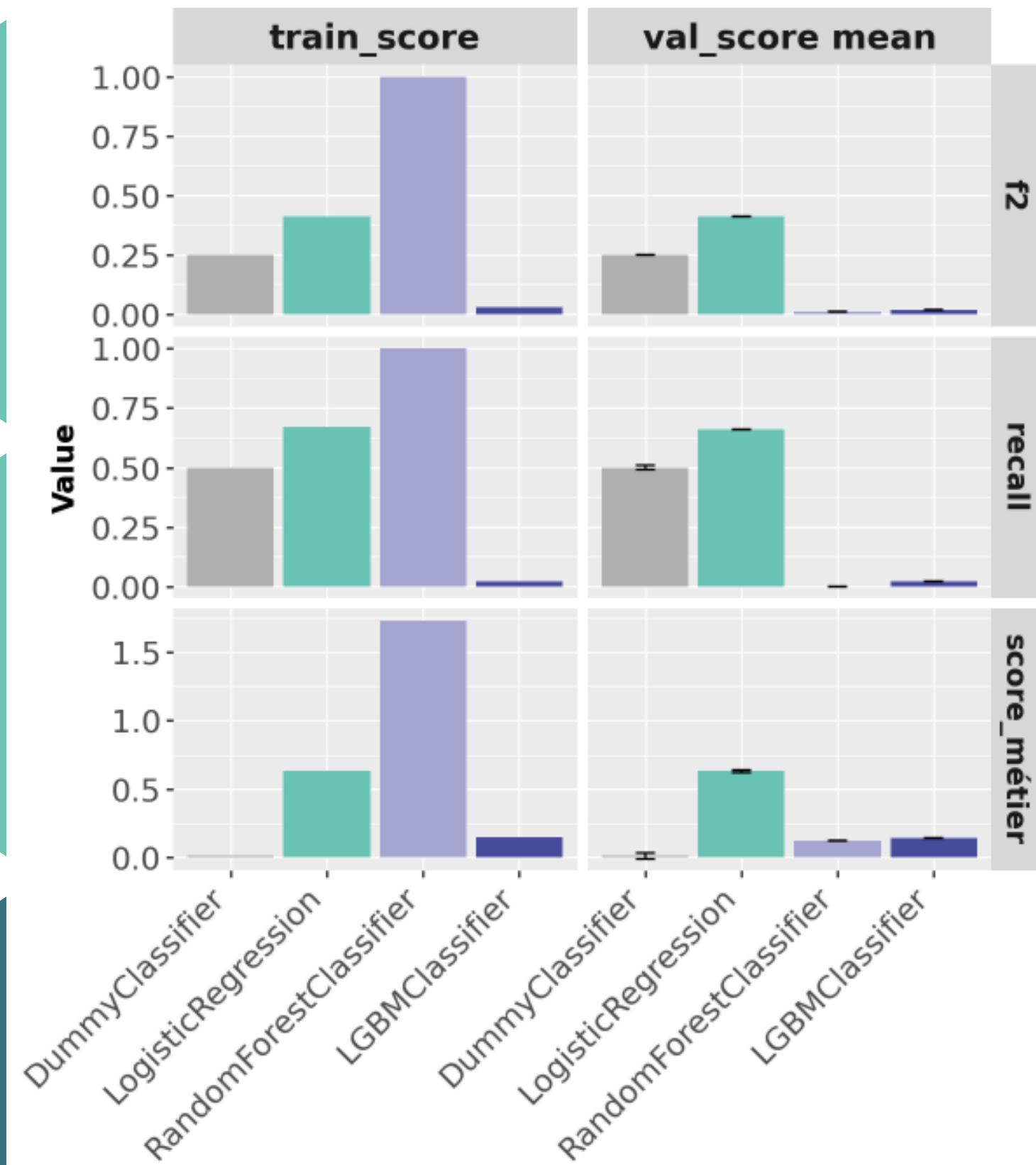
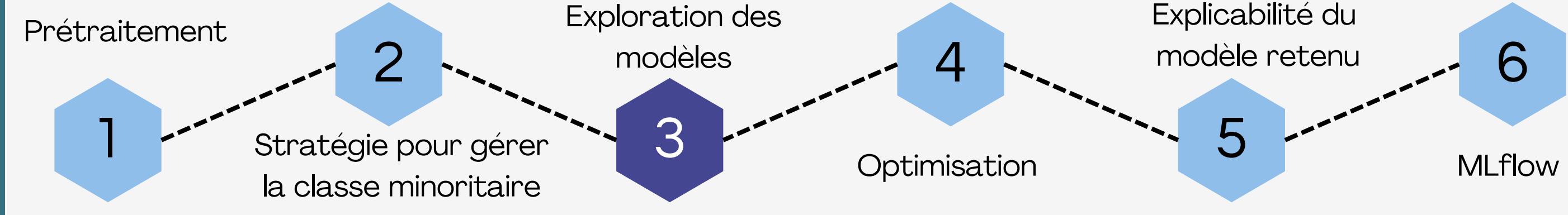
## Présentation des données

## Présentation de l'approche

## EDA + feature engineering

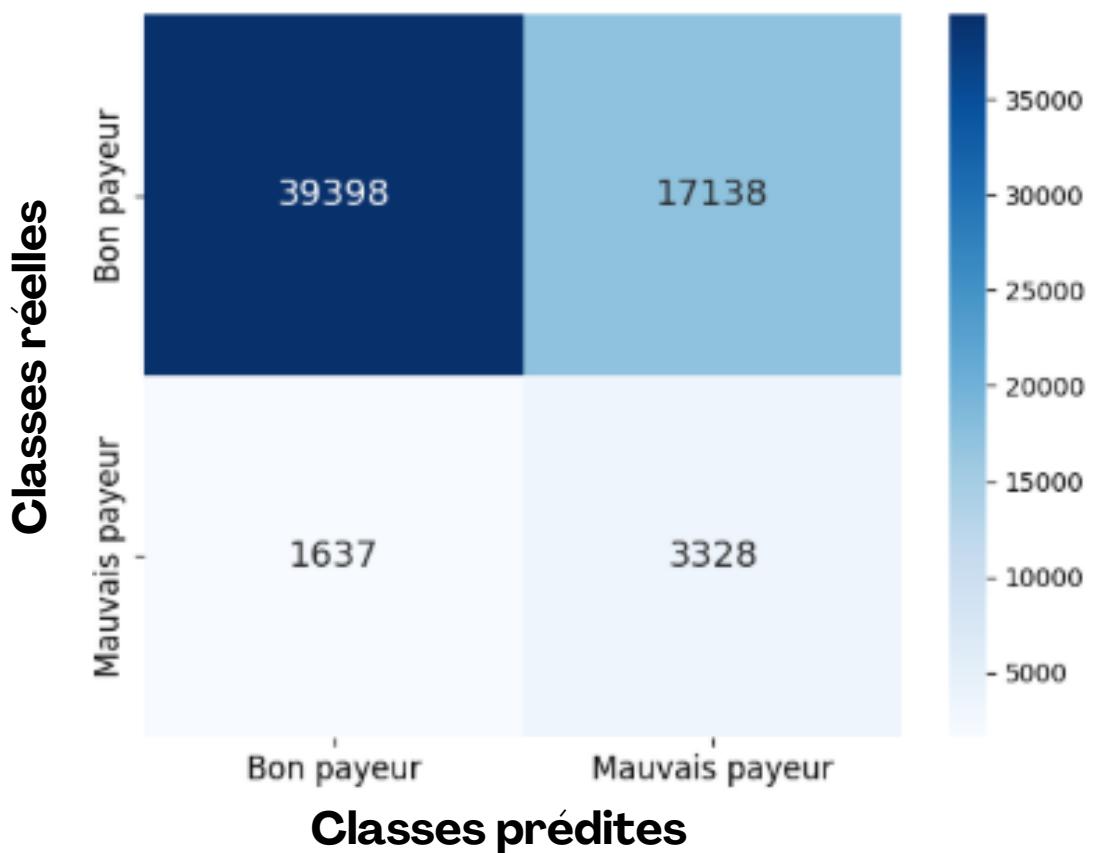
## Modélisation

## API + Dashboard



**Dummy**: rappel moyen, score métier faible  
**Reg Log**: rappel moyen, score métier > 0  
**Random Forest**: gros overfit  
**Light GBM**: rappel moyen, score métier faible

## Matrice de confusion



# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard

## Prétraitement

1

2

Stratégie pour gérer la classe minoritaire

## Exploration des modèles

3

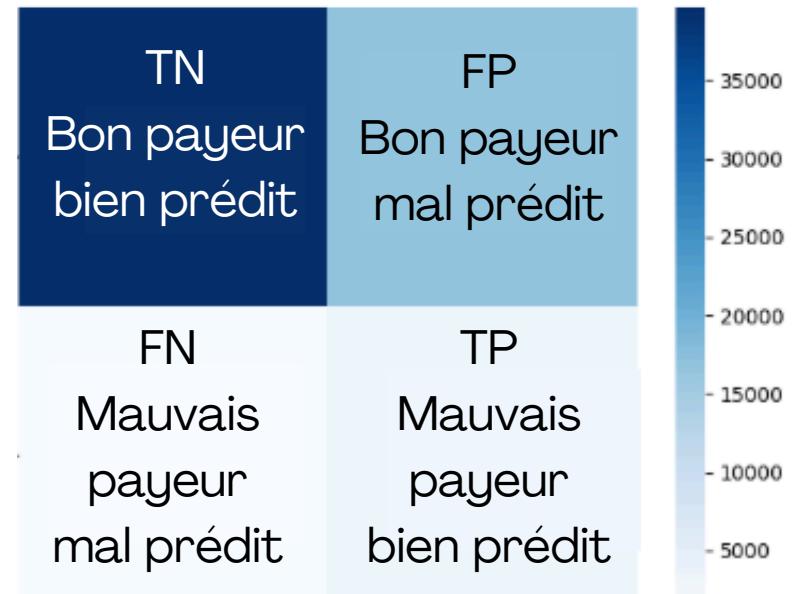
4

Optimisation

5

6

MLflow



## Score métier

Prêt: 10 000 - intérêt: 1 000

TN : + 1 000

FP : - 1 000

FN : - 10 000

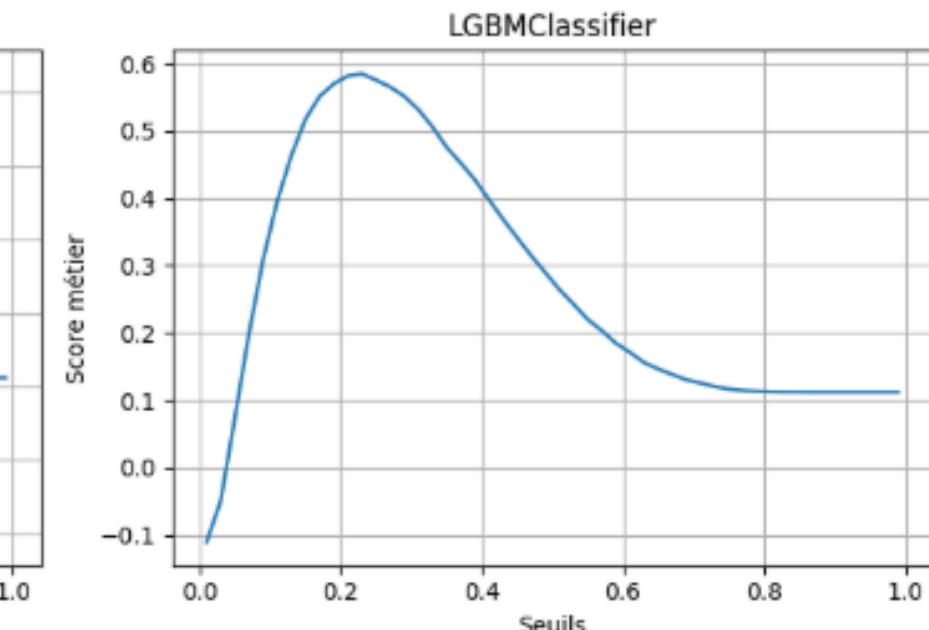
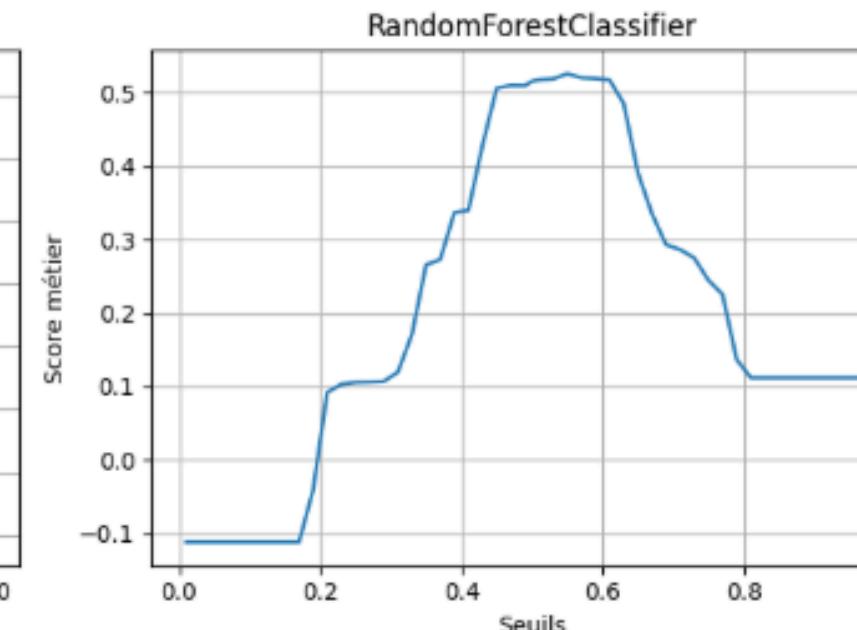
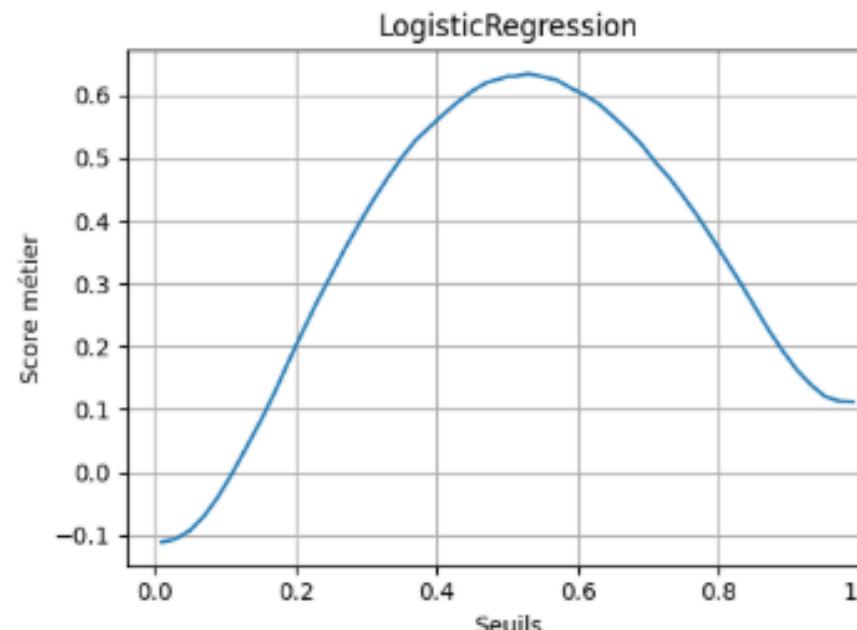
TP : + 10 000

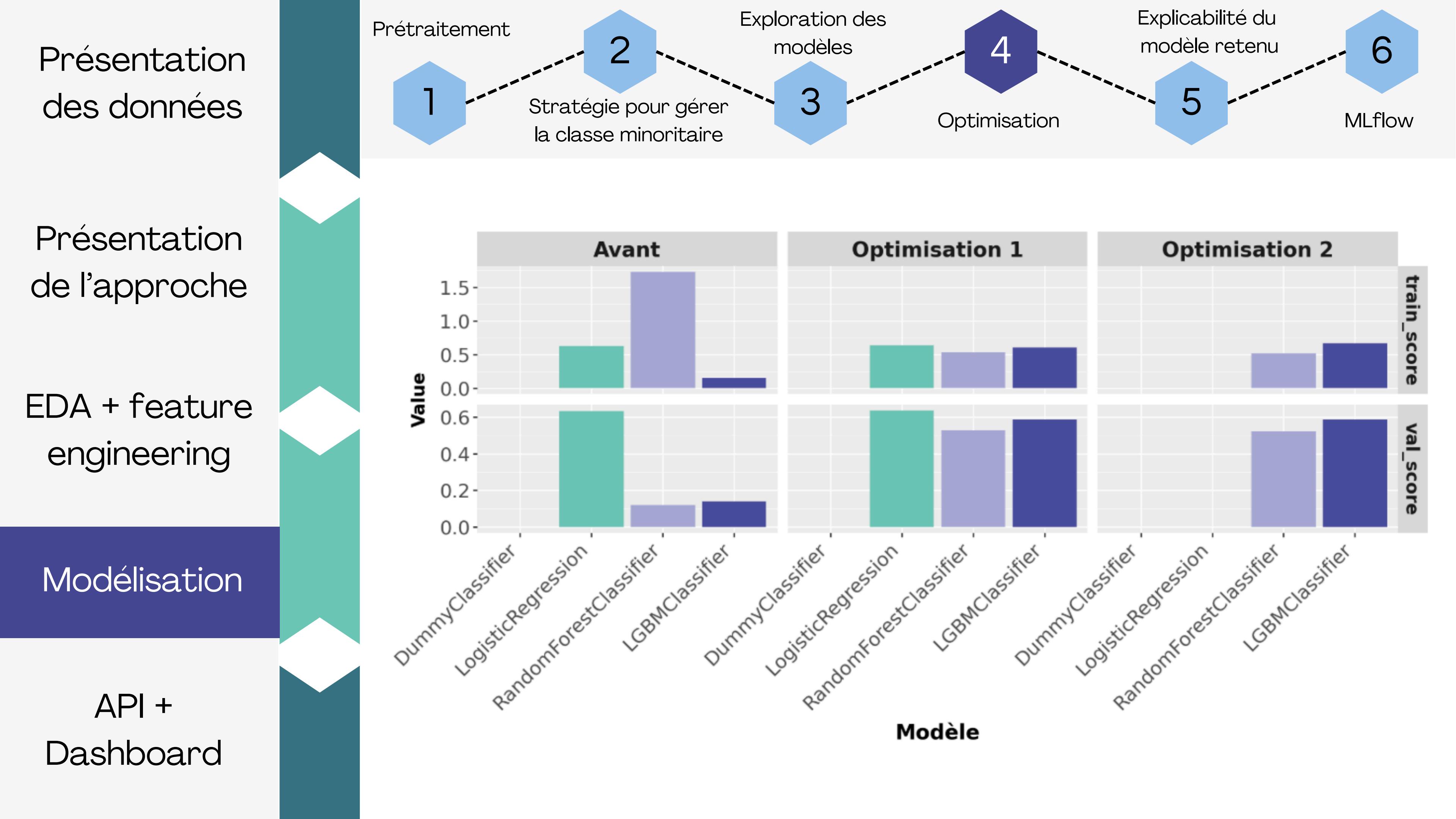
$$\text{Score} = \frac{10 * \text{TP} + \text{TN} - 10 \text{FN} - \text{FP}}{\text{TP} + \text{TN} + \text{FN} + \text{FP}}$$

## Optimisation des hyperparamètres

- Des modèles
- De la fonction SMOTE

## Optimisation du seuil de classification





## Présentation des données

## Présentation de l'approche

## EDA + feature engineering

## Modélisation

## API + Dashboard

Prétraitement

1

2

Stratégie pour gérer la classe minoritaire

Exploration des modèles

3

4

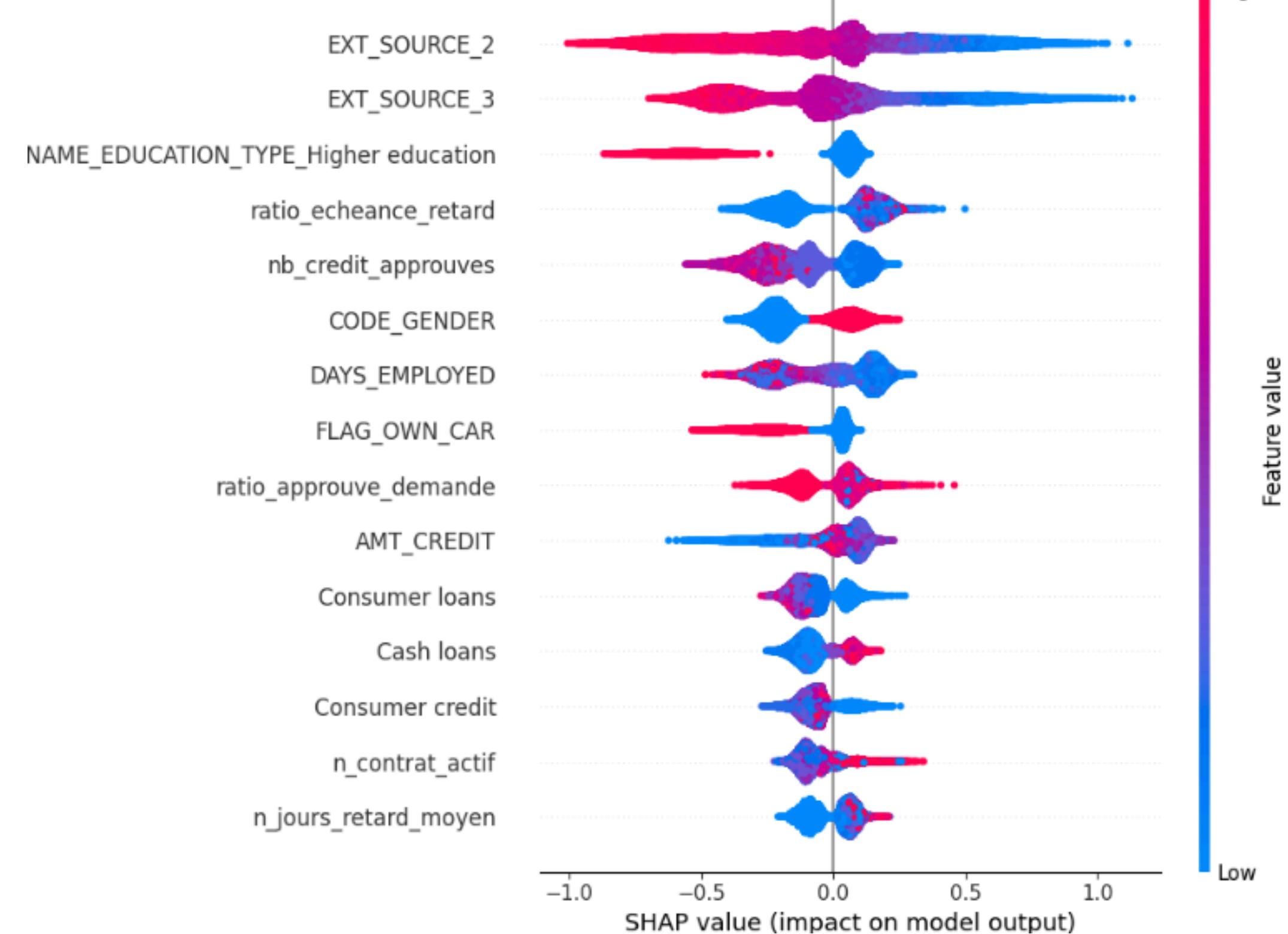
Optimisation

5

Explicabilité du modèle retenu

6

MLflow



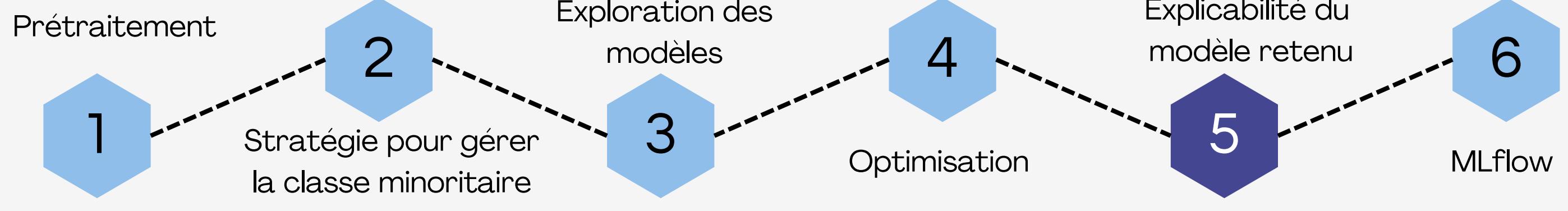
# Présentation des données

# Présentation de l'approche

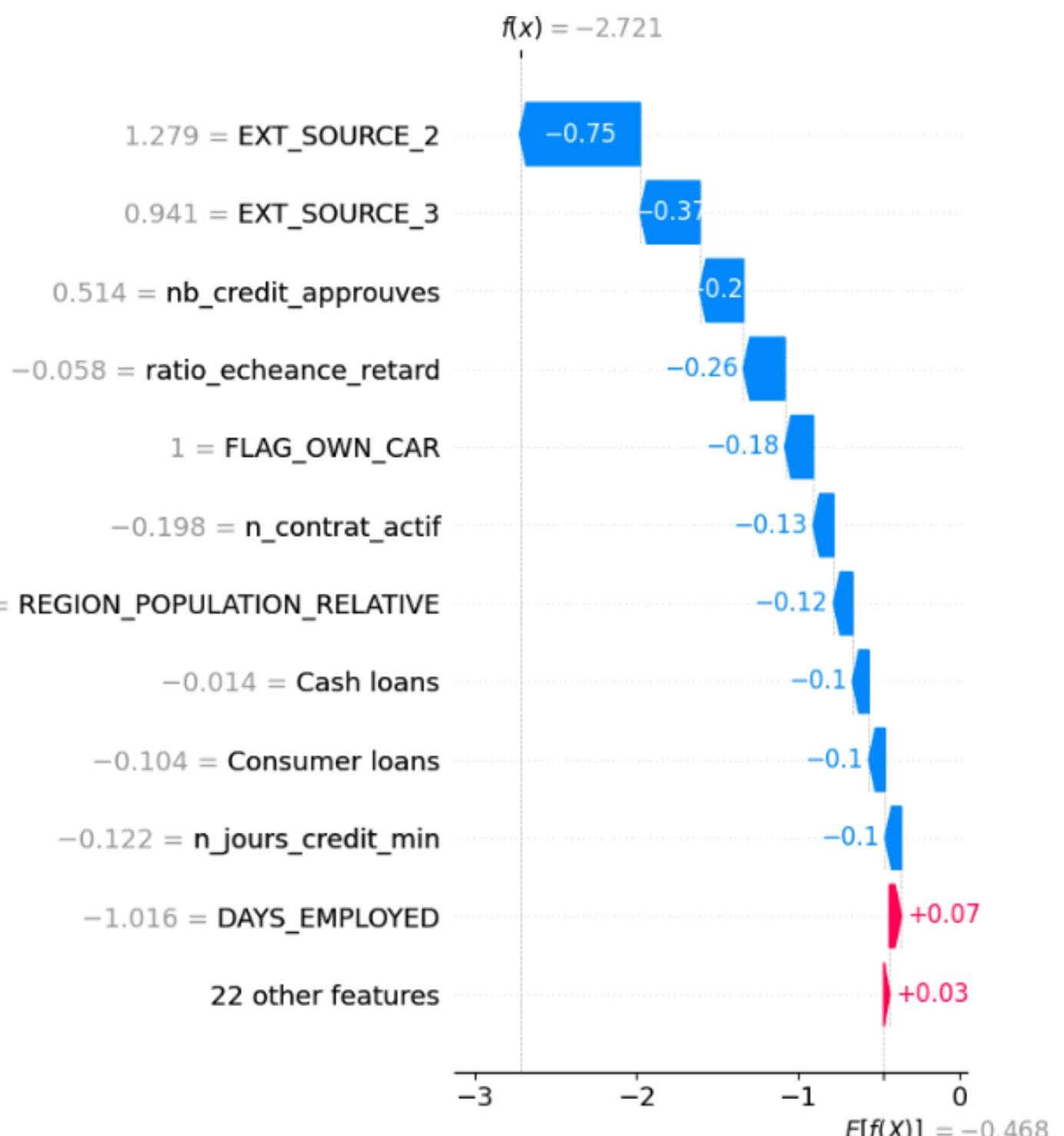
# EDA + feature engineering

# Modélisation

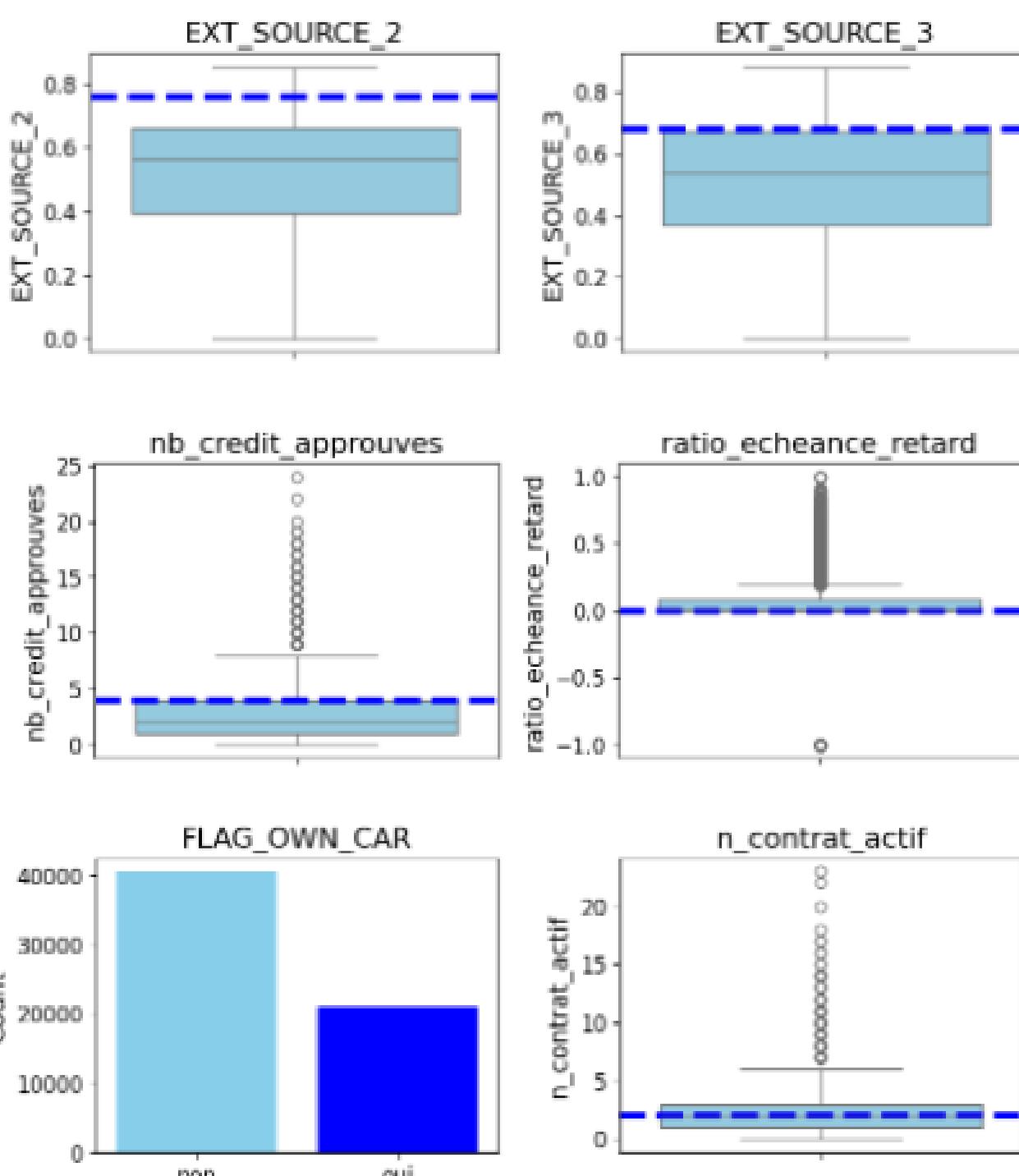
# API + Dashboard



## Explicatitivé locale



## Position de la demande par rapport à l'ensemble



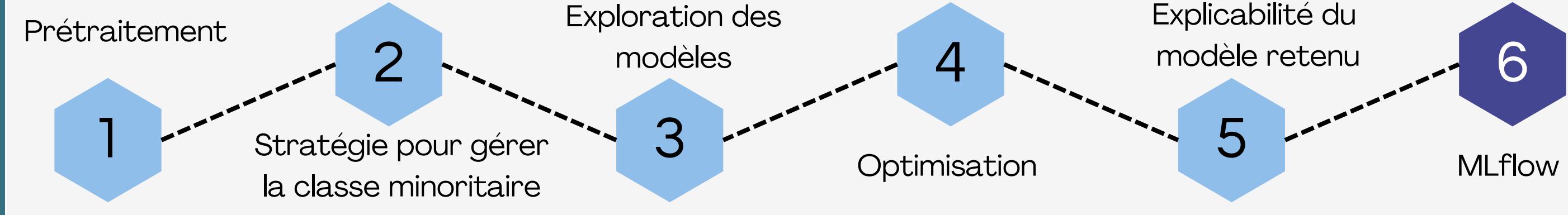
# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

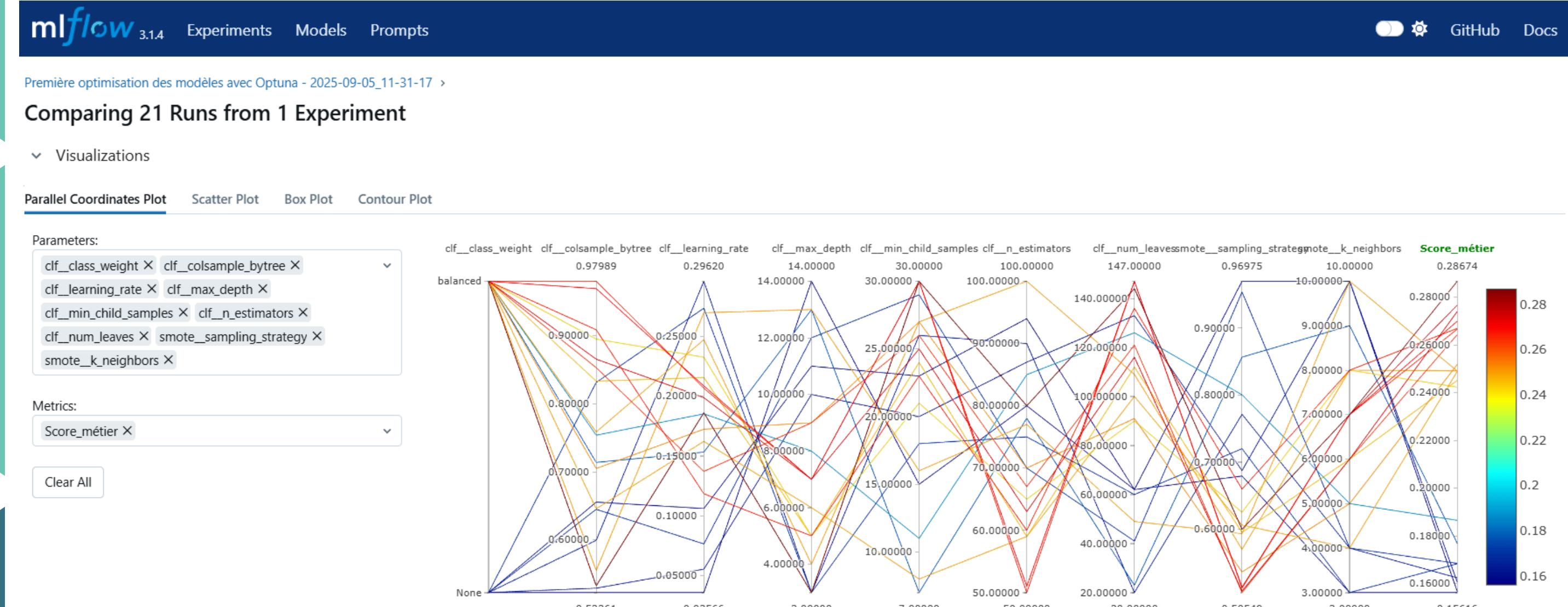
# Modélisation

# API + Dashboard



**mlflow**

## Suivi de l'optimisation



## Présentation des données

## Présentation de l'approche

## EDA + feature engineering

## Modélisation

## API + Dashboard

### Prétraitement

1

2

Stratégie pour gérer la classe minoritaire

### Exploration des modèles

3

4

Optimisation

5

### Explicabilité du modèle retenu

6

MLflow

mlflow

Sauvegarde des modèles

Sauvegarde d'artefacts

mlflow 3.1.4 Experiments Models Prompts GitHub Docs

Première optimisation des modèles avec Optuna - 2025-09-08\_21-05-33 > Models >

LGBMClassifier

Overview Traces Artifacts

MLmodel

conda.yaml  
input\_example.json  
model.pkl  
python\_env.yaml  
requirements.txt  
serving\_input\_example.json

MLmodel 7.09KB

Path: file:///C:/Users/33647/Desktop/Openclassroom/Projet7/Nouvel...

artifact\_path: file:///C:/Users/33647/Desktop/Openclassroom/Proj...  
flavors:  
python\_function:  
env:  
conda: conda.yaml  
virtualenv: python\_env.yaml  
loader\_module: mlflow.sklearn  
model\_path: model.pkl  
predict\_fn: predict  
python\_version: 3.12.3  
sklearn:  
code: null  
pickled\_model: model.pkl  
serialization\_format: cloudpickle  
sklearn\_version: 1.4.2  
is\_signature\_from\_type\_hint: false  
mlflow\_version: 3.1.4  
model\_id: m-fbac621f13d041b39c16d3e3fda04996  
model\_size\_bytes: 6880854  
model\_uuid: m-fbac621f13d041b39c16d3e3fda04996

mlflow 3.1.4 Experiments Models Prompts GitHub Docs

Optimisation modèles avec Optuna test - 2025-08-16\_10-36-40 >

Optimisation: LGBMClassifier(random\_state=42, verbosity=-1)

Overview Model metrics System metrics Traces Artifacts

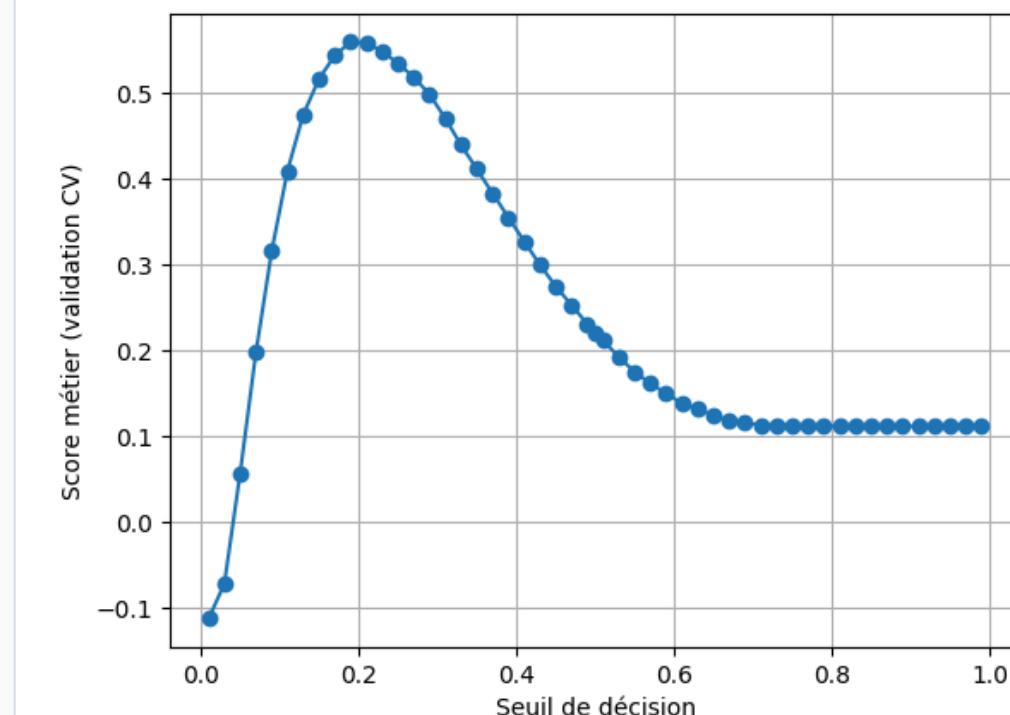
plots

LightGBM\_threshold\_optim

plots/LightGBM\_threshold\_optimization.png 30.34KB

Path: file:///C:/Users/33647/Desktop/Openclassroom/Projet7/Nouveau dossier/mlflow/mlruns/512...

Score métier en fonction du seuil de décision



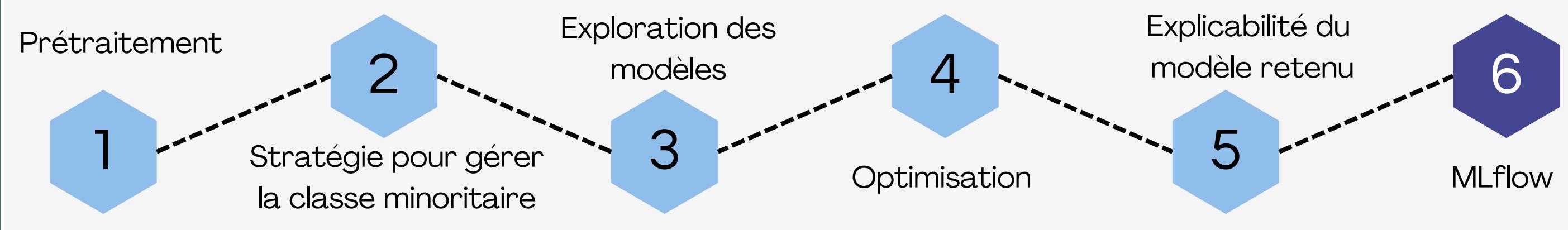
## Présentation des données

## Présentation de l'approche

## EDA + feature engineering

## Modélisation

## API + Dashboard



Métrique      PSI

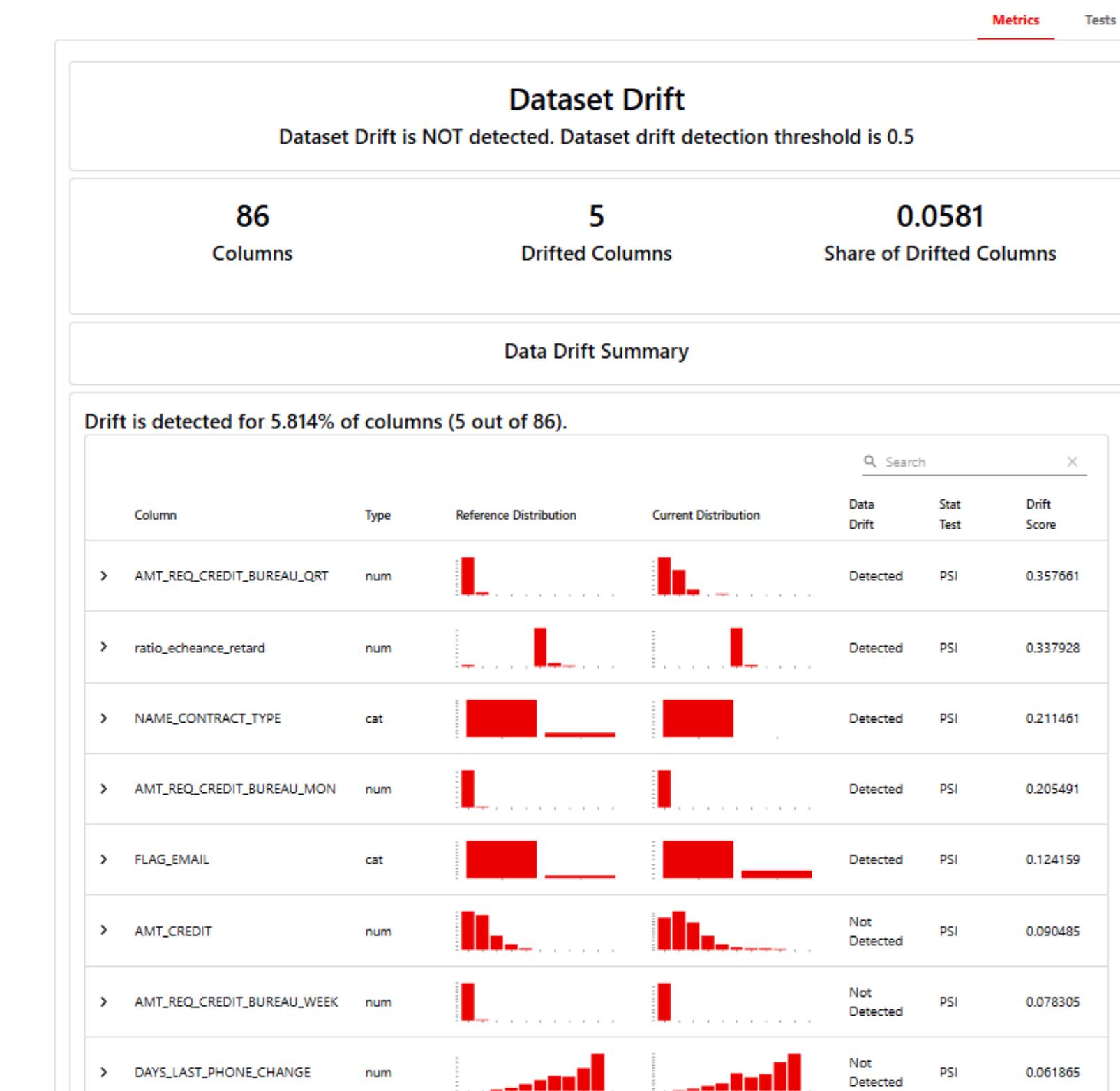
### Résultat

- 94% des variables n'ont pas dérivé
- 6% des variables ont dérivé

Variables globalement stables dans le temps



ratio\_echeance retard



# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard

Retour sur l'approche

1

2

API

Dashboard

3

4

Test unitaire

Hébergement cloud

5

6

Github

EDA  
+  
Feature engineering



**Pipeline de modélisation**

- prétraitement
- sélection de variable
- SMOTE
- modélisation

**Optimisation**  
**Modèle finale**

mlflow™

Création d'une API

FastAPI

↓  
Tests unitaires

Mise à disposition de l'outil



Création d'un Dashboard



Déploiement des applications sur le cloud

pytest



HEROKU

Render

Streamlit

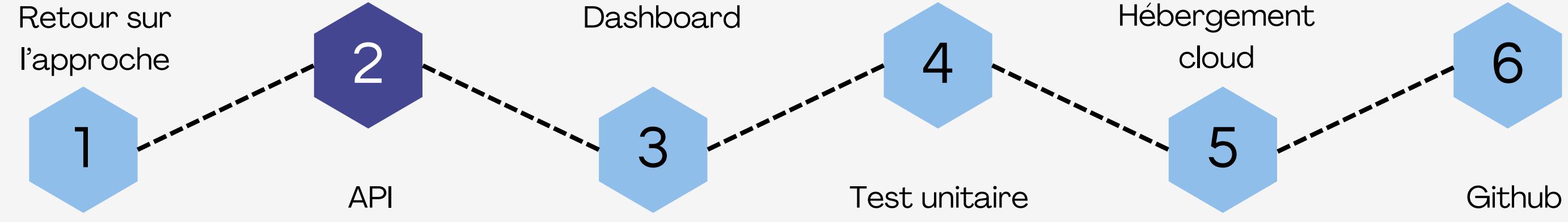
## Présentation des données

## Présentation de l'approche

## EDA + feature engineering

## Modélisation

## API + Dashboard



<https://api-oc-p7.onrender.com/docs#/>

FastAPI 0.1.0 OAS 3.1

/openapi.json

default

GET / Read Root

POST /predict Predict

Parameters

Try it out

No parameters

Request body required

application/json

Example Value | Schema

```
{  
    "client_id": 0  
}
```

**But**

Rendre l'outil accessible via l'exposition de routes

**Route /**

Affiche un message d'accueil

**Route /predict**

**Entrée**

Identifiant d'une demande

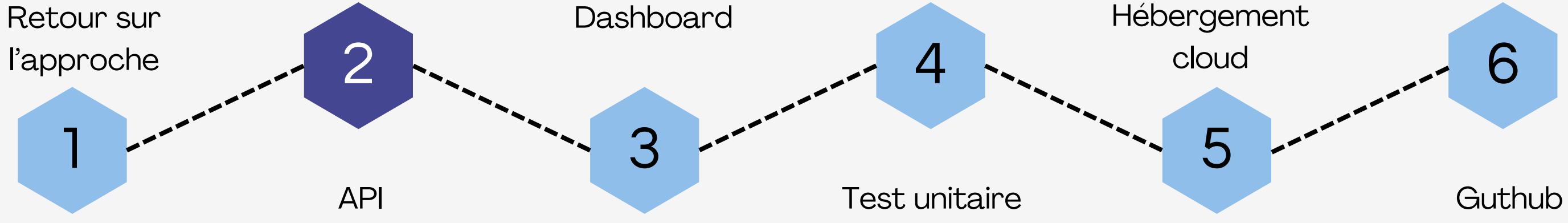
# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard



## FastAPI

<https://api-oc-p7.onrender.com/docs#/>

The screenshot shows the FastAPI documentation for the '/predict' endpoint. It includes a 'Curl' command for making a POST request to 'https://api-oc-p7.onrender.com/predict' with JSON data containing 'client\_id'. The 'Request URL' is also shown. Below, a 'Server response' section displays a 200 status code with a 'Response body' containing a JSON object with 'prediction' and 'proba' fields. The 'Response headers' section lists various HTTP headers including 'cf-cache-status', 'cf-ray', 'content-encoding', 'content-length', 'content-type', 'date', 'rndr-id', 'server', 'vary', and 'x-render-origin-server'.

### But

Rendre l'outil accessible via l'exposition de routes

### Route /

Affiche un message d'accueil

### Route /predict

#### Entrée

Identifiant d'une demande

#### Retourne

Probabilité de défaut de paiement +  
Classe Bon payeur/Mauvais payeur

Utilise une pipeline de modélisation pré entraînée et les données de la table client stockées sur github

# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard

Retour sur l'approche

1

2

API

Dashboard

3

4

Test unitaire

Hébergement cloud

5

6

Github



## Streamlit

<https://loc-p7-cu77.onrender.com>

### But

- Fournir un interface interactive et plus conviviale
- Afficher les résultats sous la forme de graphiques

### Requête vers l'API

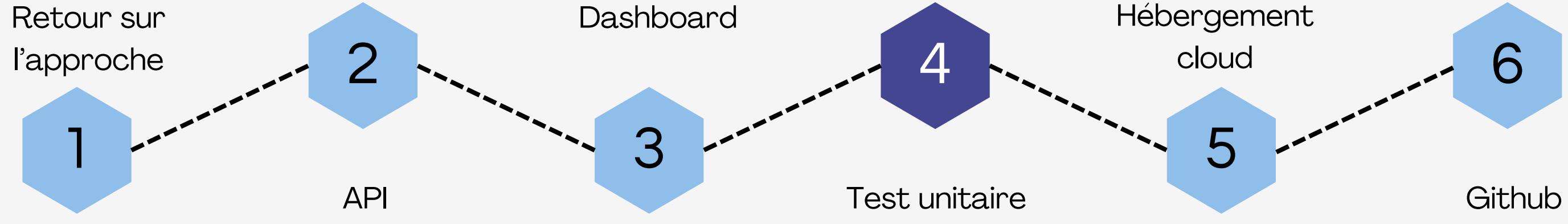
# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard



## rapport\_tests.html

Report generated on 19-Aug-2025 at 09:14:49 by [pytest-html](#) v4.1.1

### Environment

Python	3.12.3
Platform	Linux-6.11.0-1018-azure-x86_64-with-glibc2.39
Packages	<ul style="list-style-type: none"><li>• pytest: 7.4.4</li><li>• pluggy: 1.6.0</li></ul>
Plugins	<ul style="list-style-type: none"><li>• metadata: 3.1.1</li><li>• anyio: 4.10.0</li><li>• html: 4.1.1</li></ul>
CI	true
JAVA_HOME	/usr/lib/jvm/temurin-17-jdk-amd64

### Summary

4 tests took 75 ms.

(Un)check the boxes to filter the results.

<input checked="" type="checkbox"/> 0 Failed,	<input checked="" type="checkbox"/> 4 Passed,	<input checked="" type="checkbox"/> 0 Skipped,	<input checked="" type="checkbox"/> 0 Expected failures,	<input checked="" type="checkbox"/> 0 Unexpected passes,	<input checked="" type="checkbox"/> 0 Errors,	<input checked="" type="checkbox"/> 0 Reruns	<a href="#">Show all details</a>
Result ▲	Test						Duration
Passed	Test/test_api.py::test_read_root						55 ms
Passed	Test/test_api.py::test_predict_success						15 ms
Passed	Test/test_api.py::test_predict_client_not_found						2 ms
Passed	Test/test_api.py::test_predict_validation_error						3 ms

### But

Vérifier que l'API fonctionne comme attendue



pytest

### Tests

### Route d'entrée

Vérification du message d'accueil

### Route prédict

### Vérification

- Modèle d'entrée
- Client non valide
- Client valide
  - réponse
  - type probas

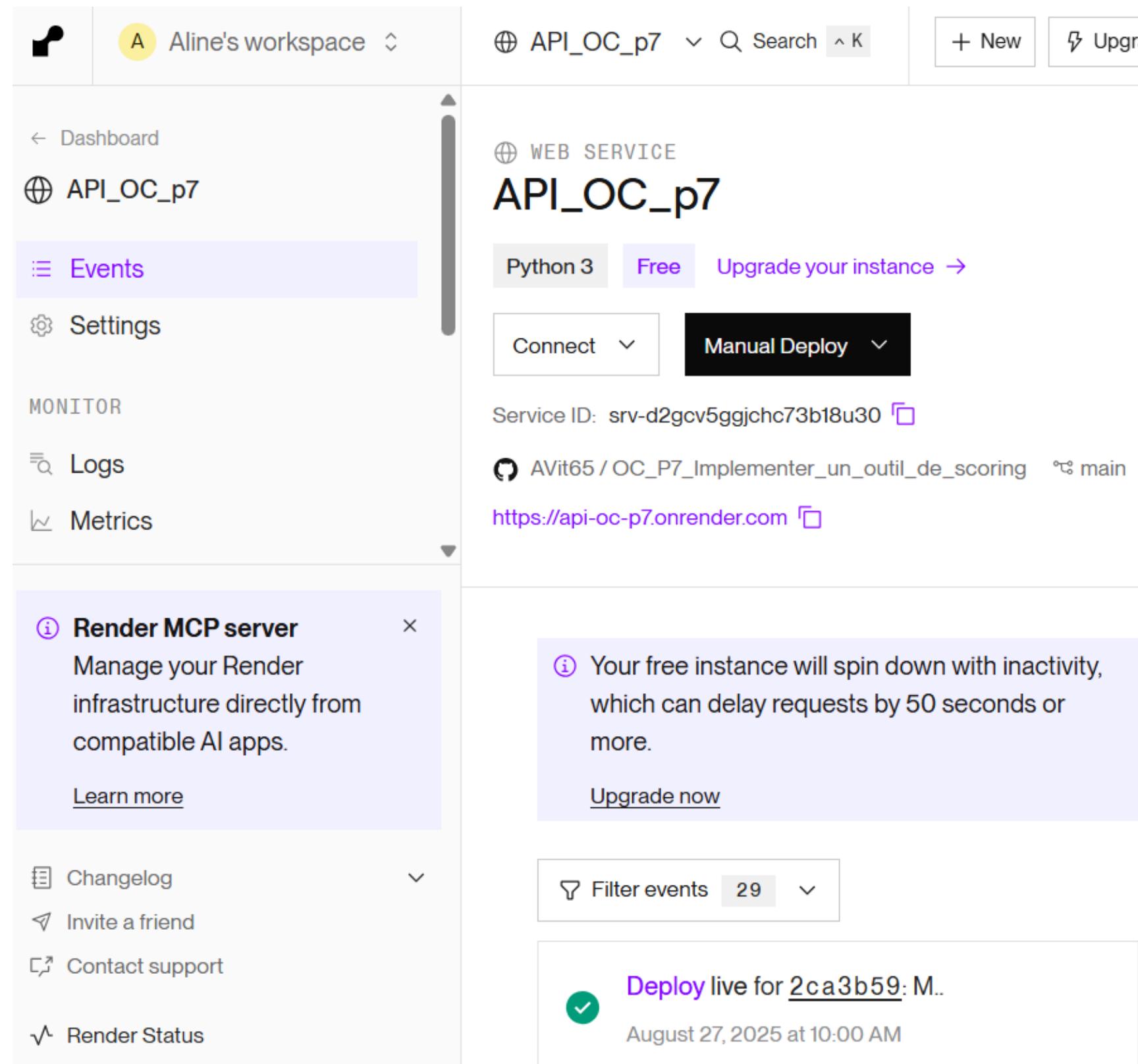
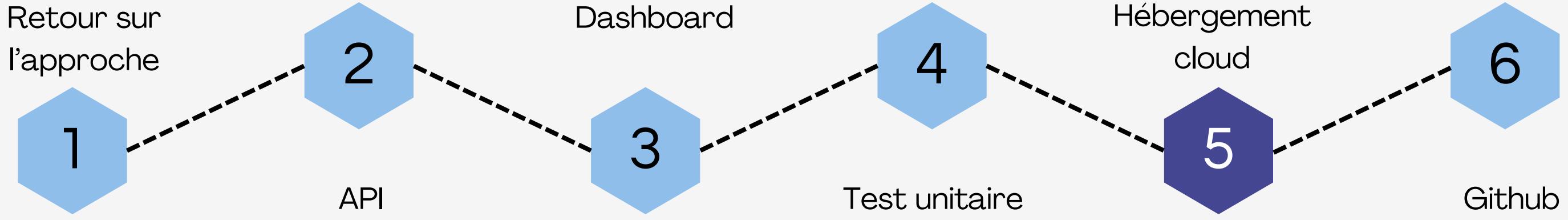
# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard



## Render

### But

Héberger l'API et le dashboard sur le cloud

→ deux services à configurer

## Configuration

- Relier au repo Github
- déploiement a chaque commit
- Commande pour lancer l'app
- Commande pour construire l'environnement

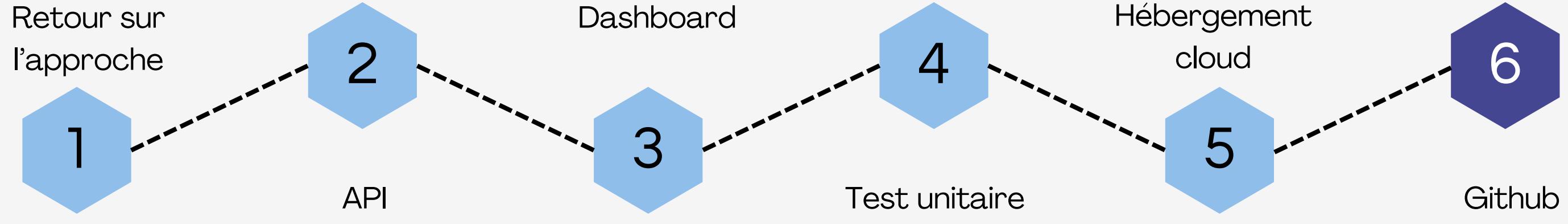
# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard

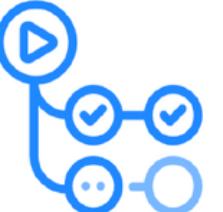


[https://github.com/AVit65/OC\\_P7\\_Implementer\\_un\\_outil\\_de\\_scoring](https://github.com/AVit65/OC_P7_Implementer_un_outil_de_scoring)

The screenshot shows a GitHub repository page for 'AVit65 / OC\_P7\_Implementer\_un\_outil\_de\_scoring'. The repository is public and has 0 stars, 0 forks, and 0 watching. It contains 6 commits, including modifications to .github/workflows, API, Config, Images, Notebook, Output, Streamlit, Test, Utils, .gitignore, .python-version, and Readme.md. The repository has no releases or packages published. The Languages section shows Jupyter Notebook (60.9%), HTML (39.0%), and Python (0.1%).



## Github Action



## But

- Héberger le code source et les données
- Gérer les versions du code
- Automatiser des workflows

## Doit contenir

- Le code et les data
- Un fichier Requirements
- Un fichier Readme

Nécessite une architecture organisée

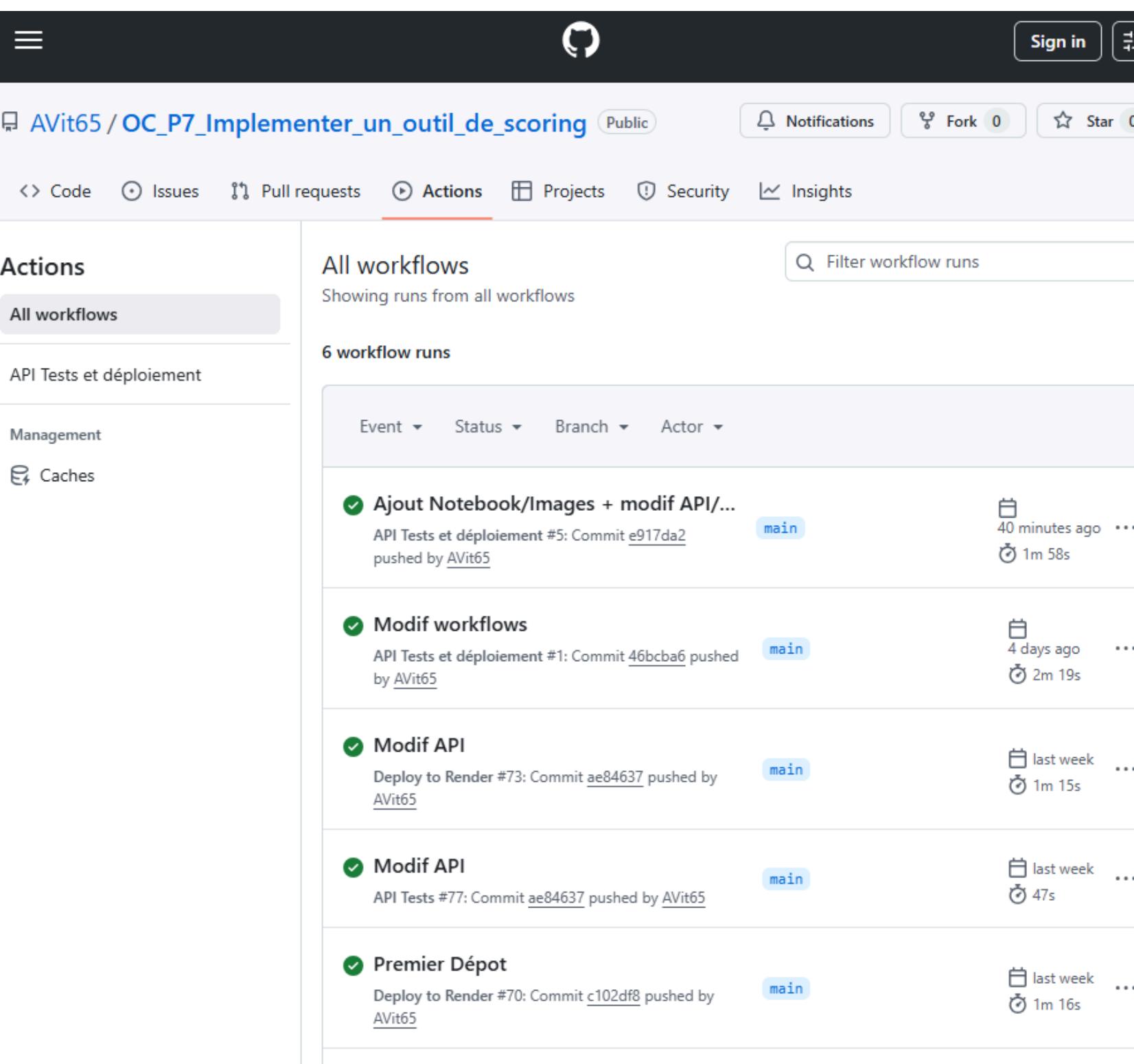
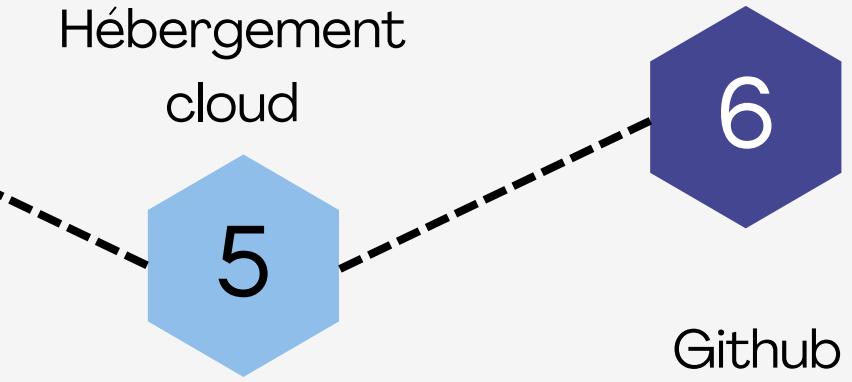
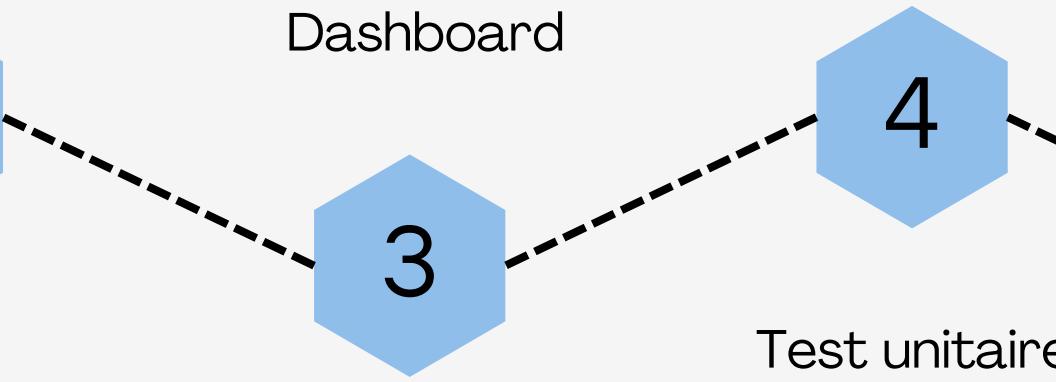
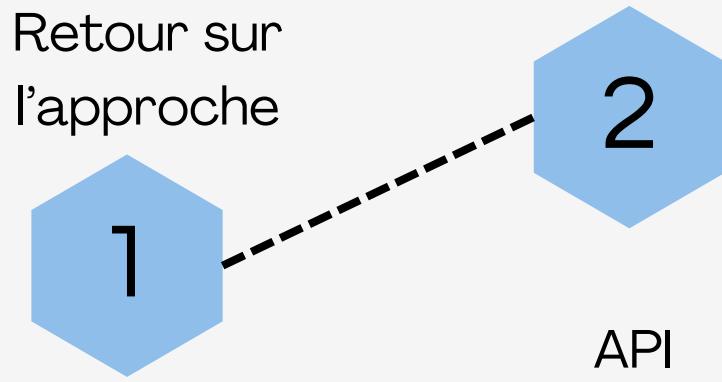
# Présentation des données

# Présentation de l'approche

# EDA + feature engineering

# Modélisation

# API + Dashboard



## Github Action



## But

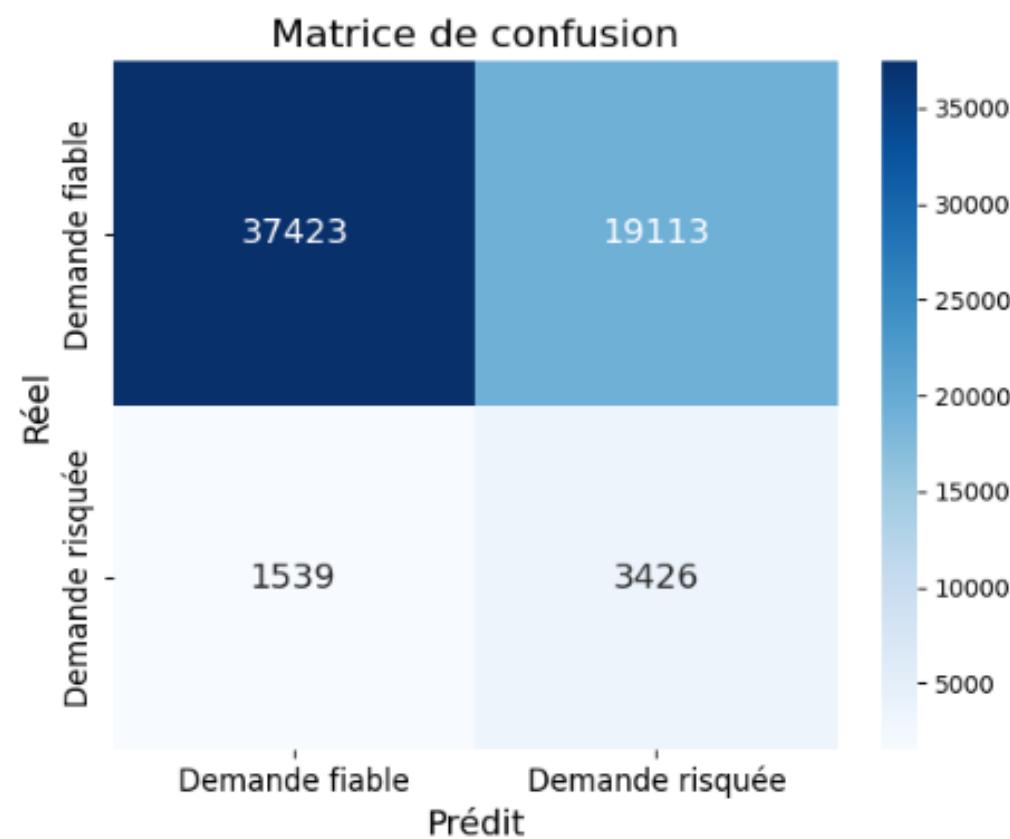
- Héberger le code source et les données
- Gérer les versions du code
- Automatiser des workflows

## Workflows

- Test automatique de l'API à chaque push
- Déploiement de l'API et du dashboard automatiquement à chaque push

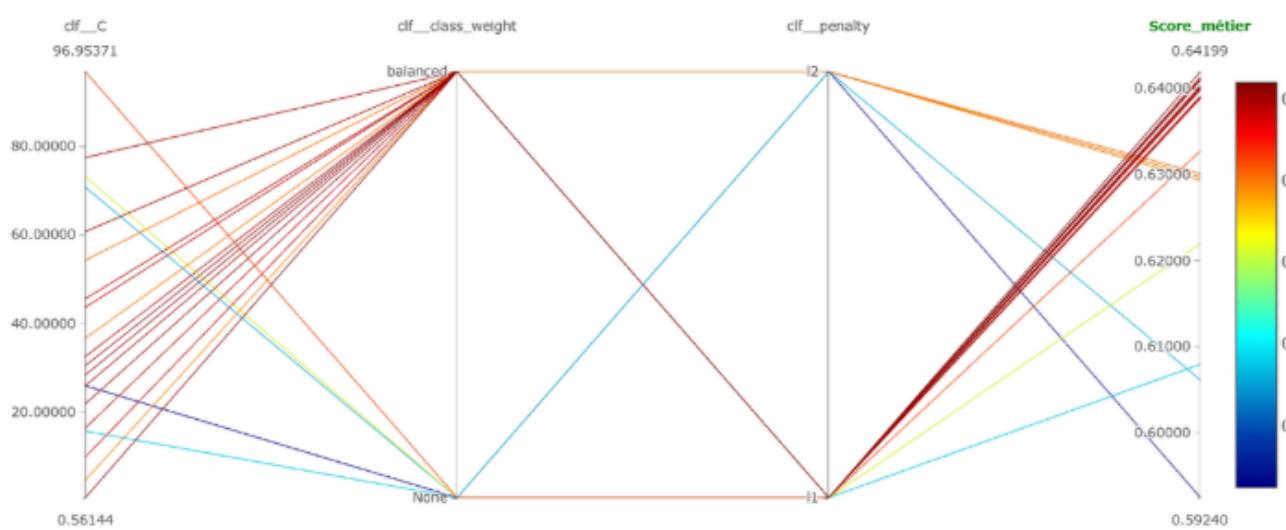
# Conclusions

Implementer un outil de scoring



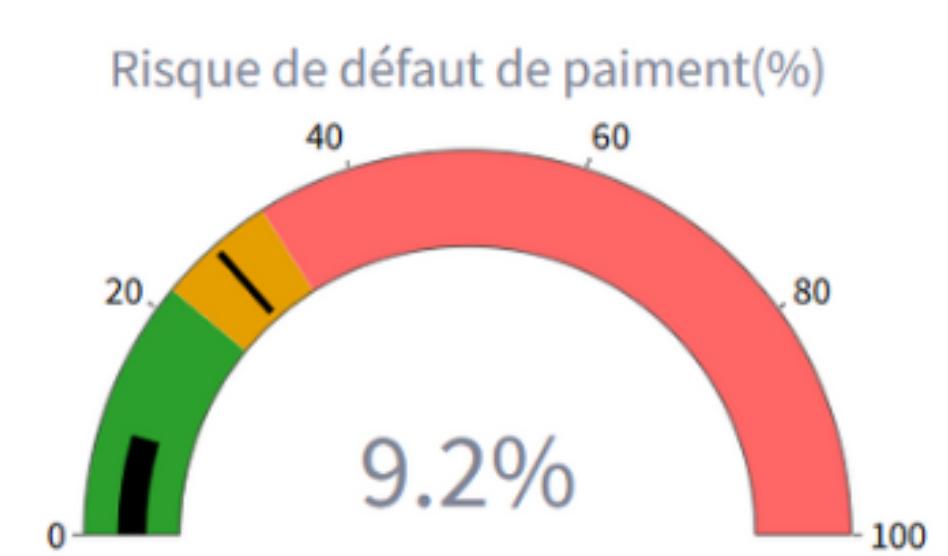
Suivre l'optimisation via MLFlow

- mlflow
- Suivie de l'optimisation
  - Sauvegarde des modèles



Deployer le modèle dans le cloud

- Développement d'un API
- Développement d'un dashboard



Merci pour votre attention