P7 - Implémentez un modèle de scoring

Deployment of a scoring API on internet

Summary

- Preliminary
- Exploratory Analysis
- Modelization
- Recording experiments and models with MLFlow
- API with FASTAPI and dashboard with streamlit
- Testing with Pytest
- Sharing code and versioning with Github
- Deployment of the API on internet with Heroku
- Drift analysis
- Conclusion

Preliminary

- **Mission :** deploy on internet a **classification algorithm** to predict a **solvability probability** for potential customers of « **Prêt à dépenser** » applying for a loan.
- **Requirements :** ensure the **transparency** of the classification for the customers on an **interactive dashboard.**

Exploratory Data Analysis

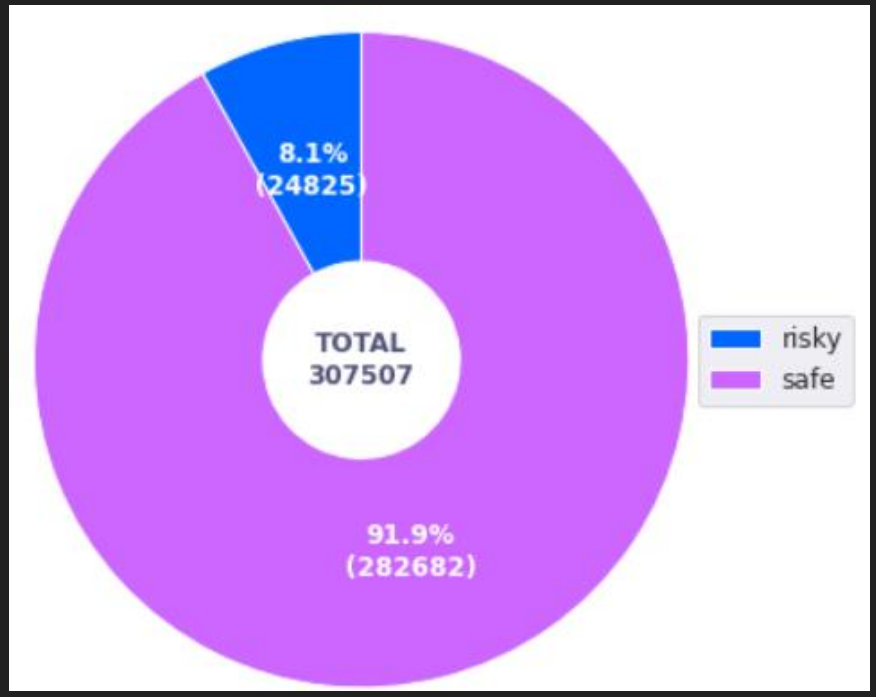
- Data are dispatched among ten files :
  - HomeCredit_columns_description.csv
  - POS_CASH_balance.csv
  - application_test.csv
  - application_train.csv
  - bureau.csv
  - bureau_balance.csv
  - credit_card_balance.csv
  - installments_payments.csv
  - previous_application.csv
  - sample_submission.csv

Exploratory Data Analysis

- As suggested, we use the cleaning and merging job done on :
  - https://www.kaggle.com/code/jsaguiar/lightgbm-with-simple-features/script
- The final resulting dataset :
  - 506 features
  - 356251 rows :
    - 307507 for modelization
    - 48744 for new customers

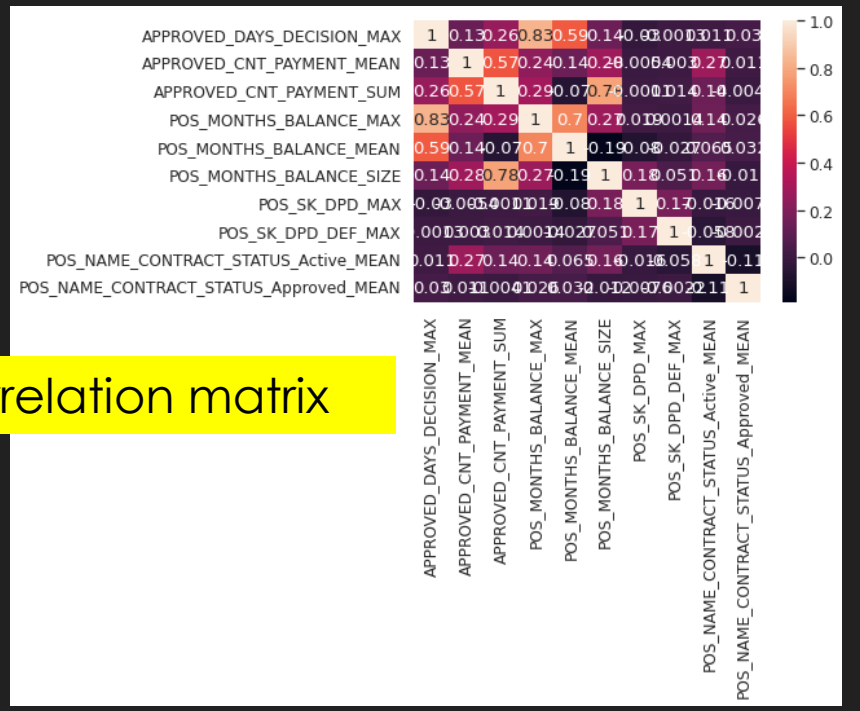
Exploratory Data Analysis

- We check how balanced the data for modelization are:



Exploratory Data Analysis

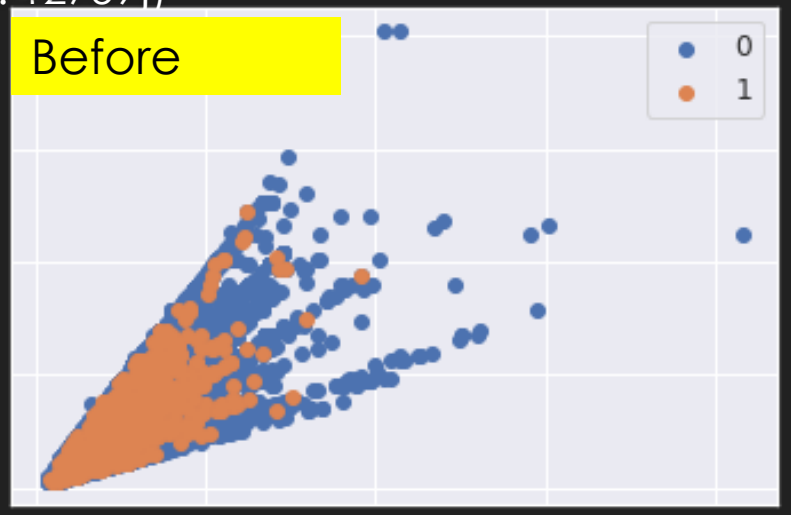
- Further, we remove data_leaks
  - features EXT_SOURCE : scores highly related with TARGET
- And we remove features contributing to bivariate correlations above 0.90.
  - 506 features => 434 features

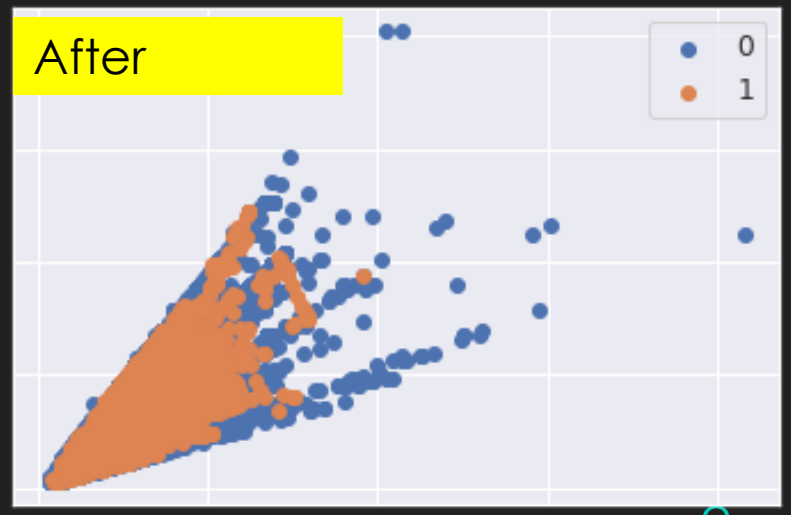


Extract of the correlation matrix

Modelization

- Balancing data with BorderlineSMOTE
  - *BorderlineSMOTE is a data augmentation technique used in machine learning to balance imbalanced datasets by generating synthetic minority samples .*
  - Before : Counter({0: 12759, 1: 1116})*
  - After : Counter({0: 12759, 1: 12759})*





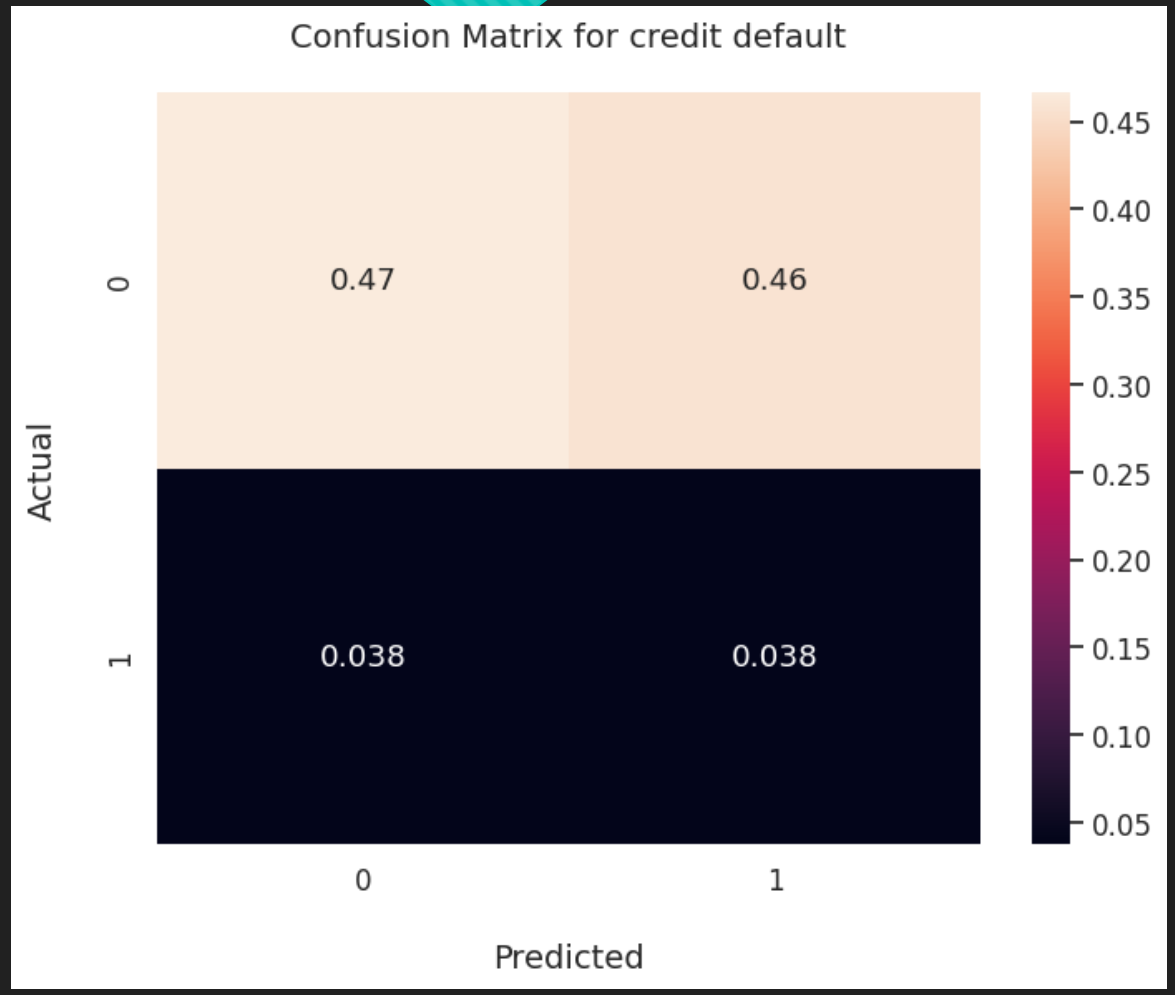
** "0" stands for customer with no solvability risk and "1" stands for customer with a solvability risk*

Modelization

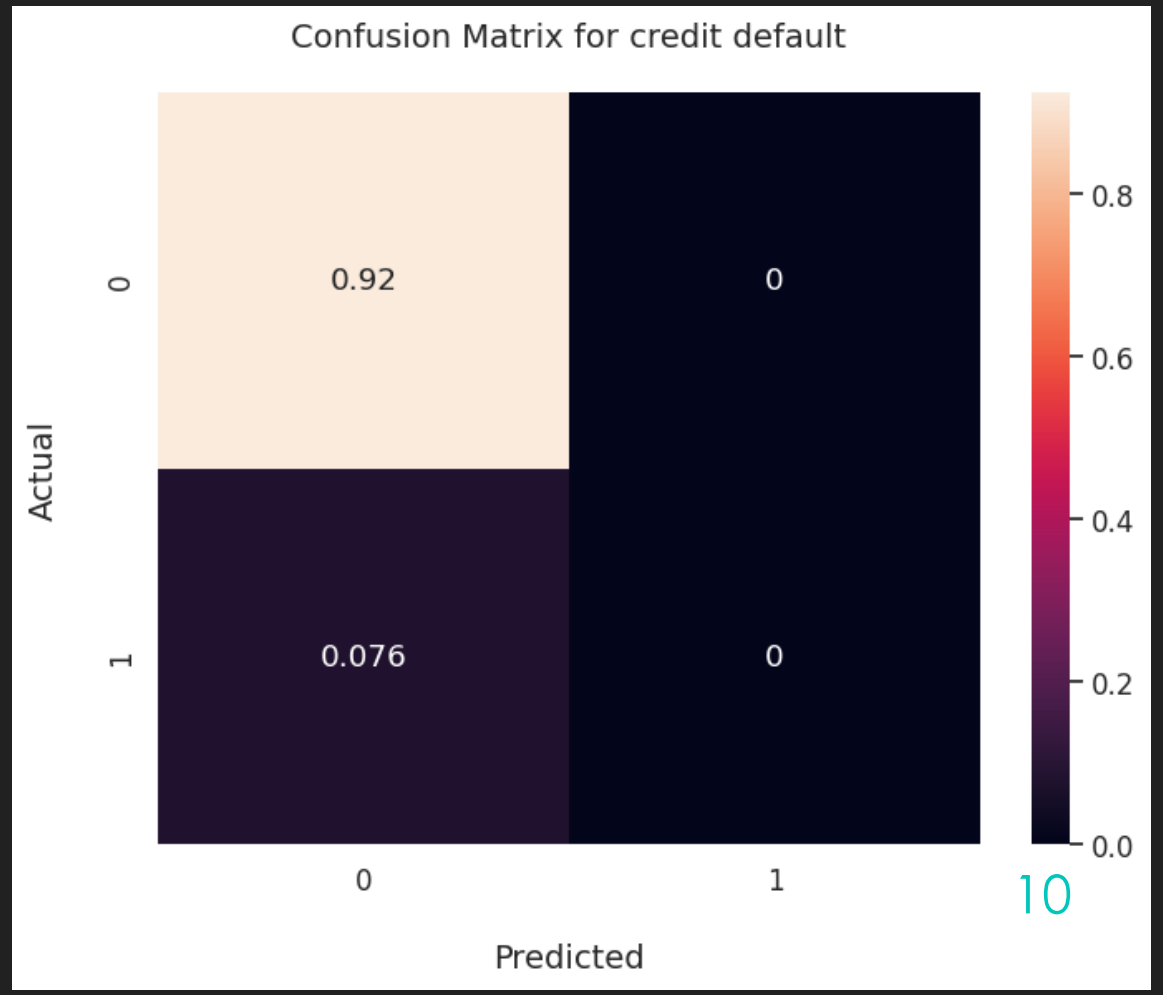
- ***RandomizedSearch*** *is a hyperparameter tuning technique in machine learning that randomly samples from a defined search space to find the optimal combination of hyperparameters for a given model*
- We cross validate four models with RandomizedSearch :
  - `DummyClassifier`
  - `LogisticRegression`
  - `RandomForestClassifier`
  - `LGBMClassifier`
- We select the best model according to its minimal custom metric on the test sample:
  - **Custom Metric = 10FN + FP**

Modelization

DummyClassifier

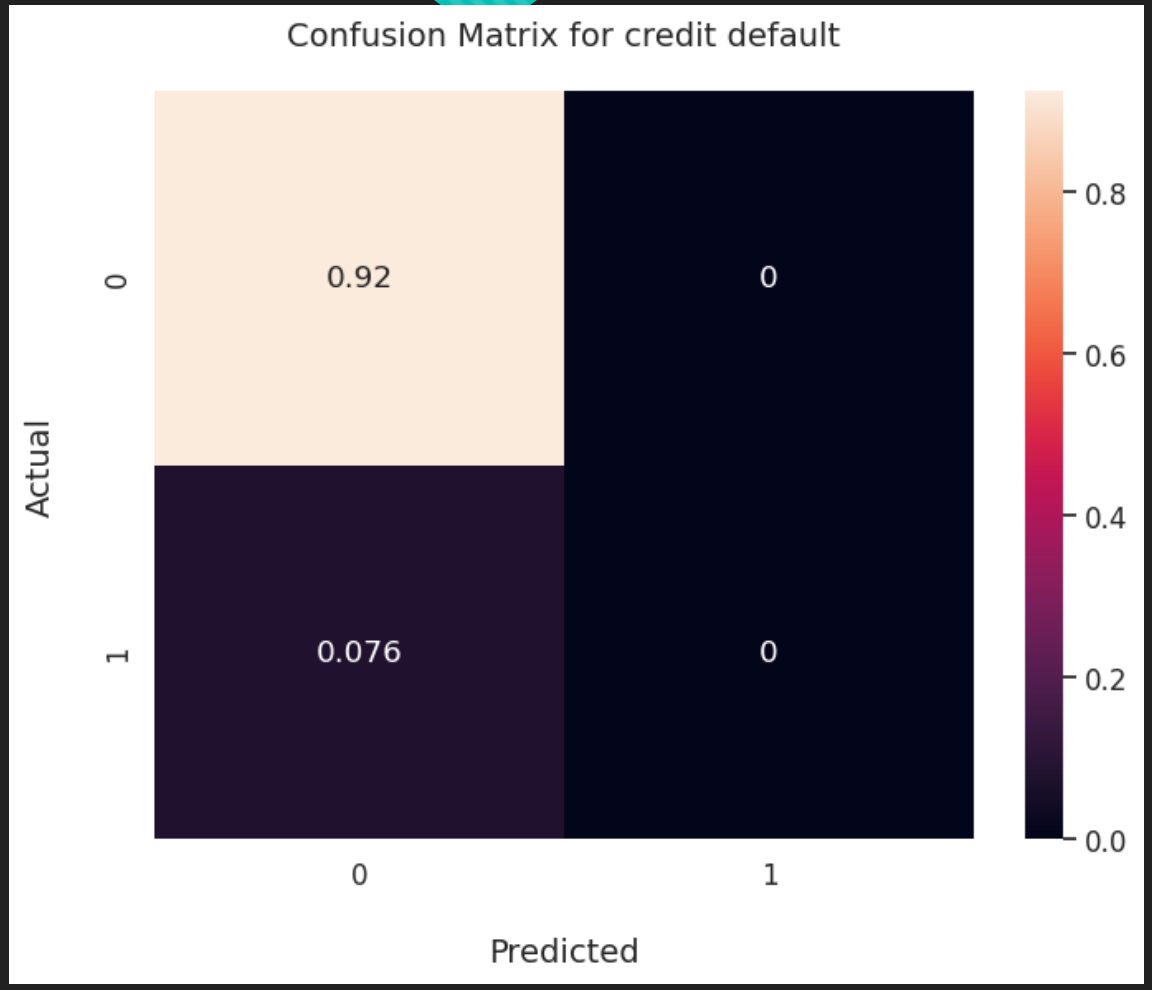


LogisticRegression

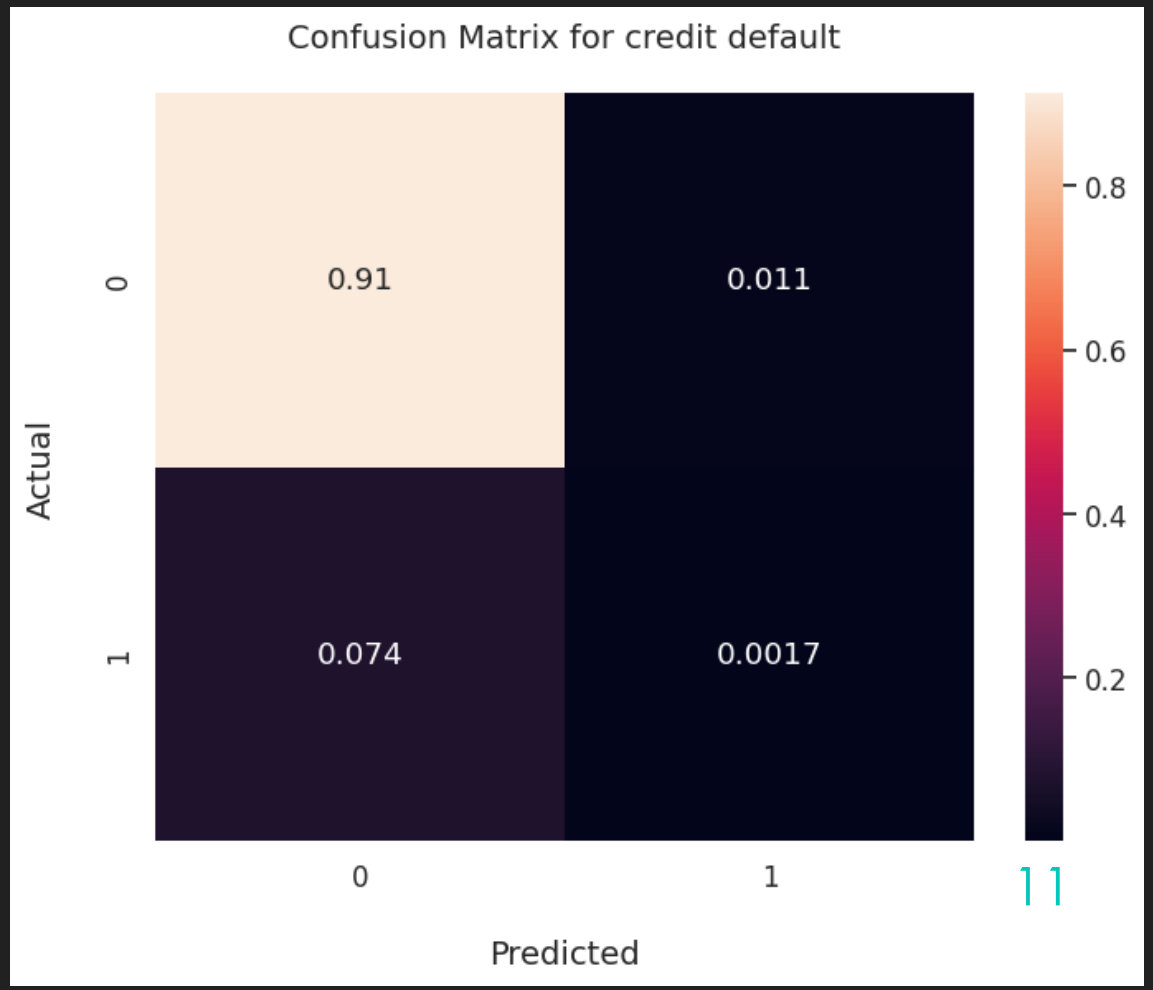


Modelization

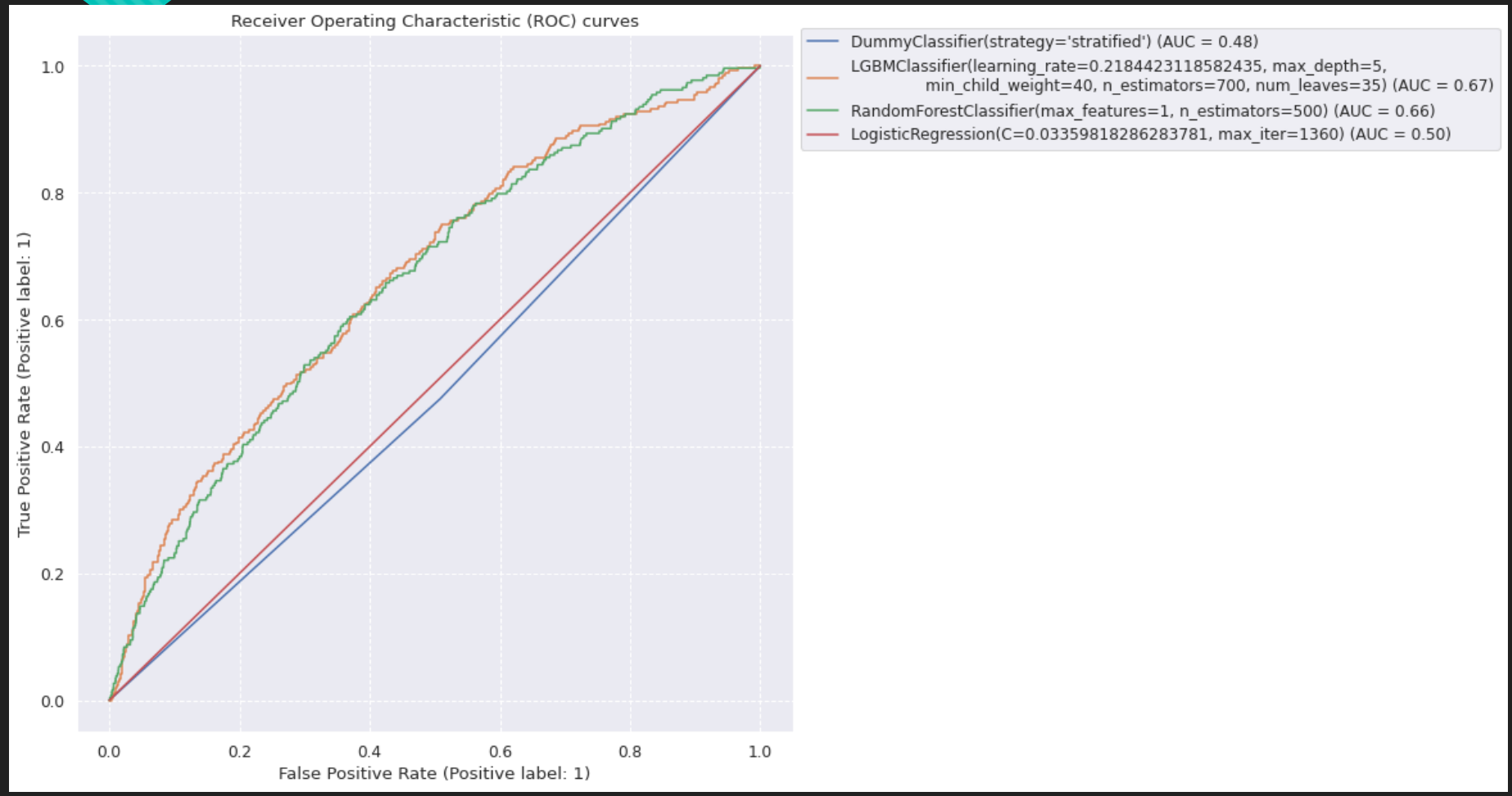
RandomForestClassifier



LGBMClassifier



Modelization



Modelization

models	Custom Cost train	Custom Cost validation	Accuracy
DummyClassifier(strategy ='stratified')	2.75	0.837994	0.504468
LGBMClassifier(learning_rate=0.2184423118582 43...	0.41	**0.751802**	0.914961
(DecisionTreeClassifier(max_features=1, random...	0.37	0.758144	0.924186
LogisticRegression(C=0.03359818286283781, max_...	5.00	0.758144	0.924186

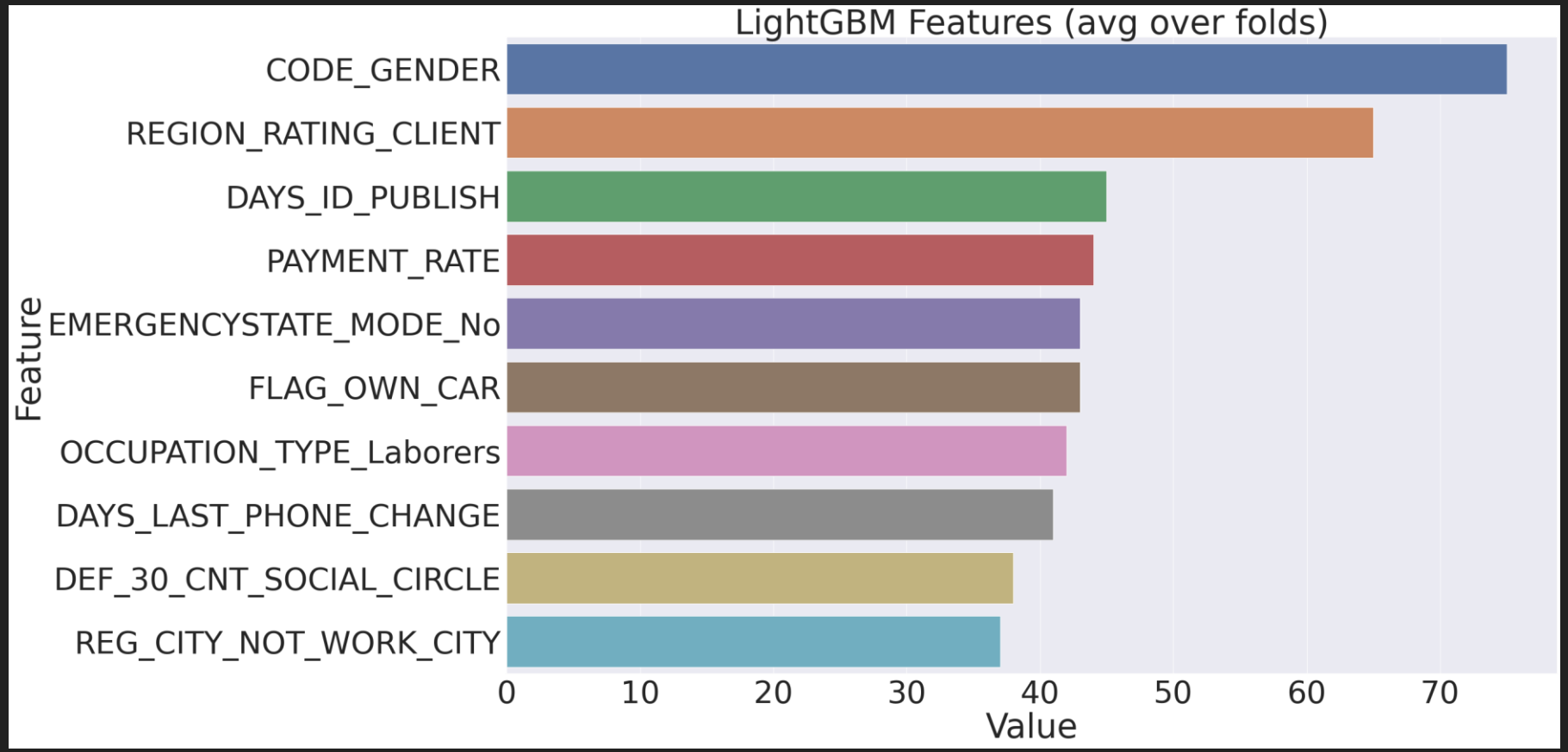
- **Best model : LGBMClassifier**

Modelization

- The score obtained in the previous table are for a probability threshold of 0.5
- Now we measure the score for 100 threshold values between 0 and 1
- Probality threshold optimization :
  - Best Cost on test sample : 0.626
  - **Best Threshold : 0.09**

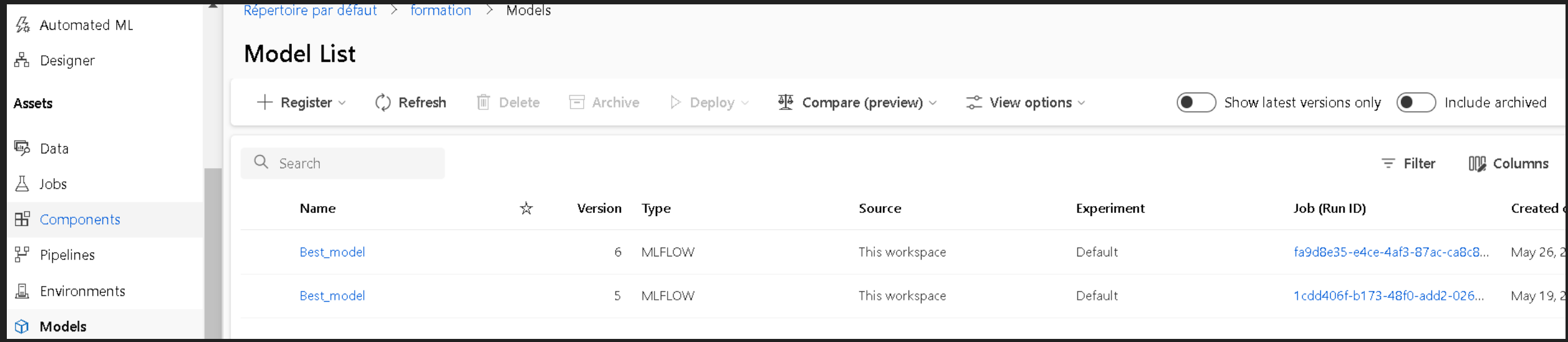
Modelization

- Feature importance



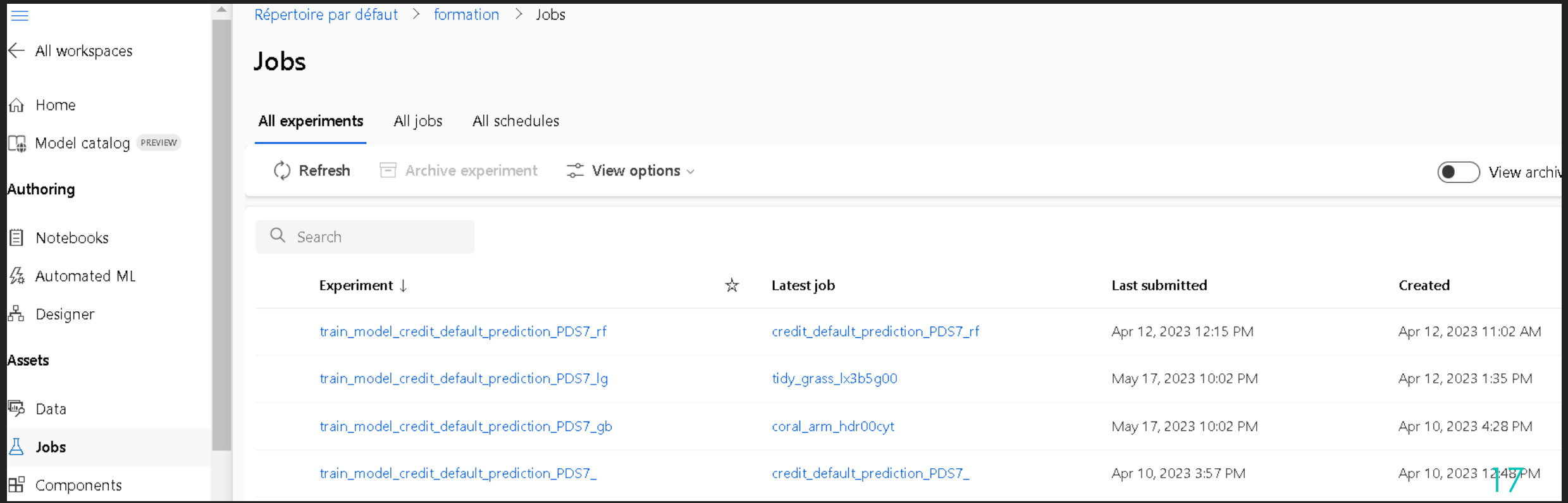
Recording experiments and models with MLFlow

- We record models in Microsof Azure studio :

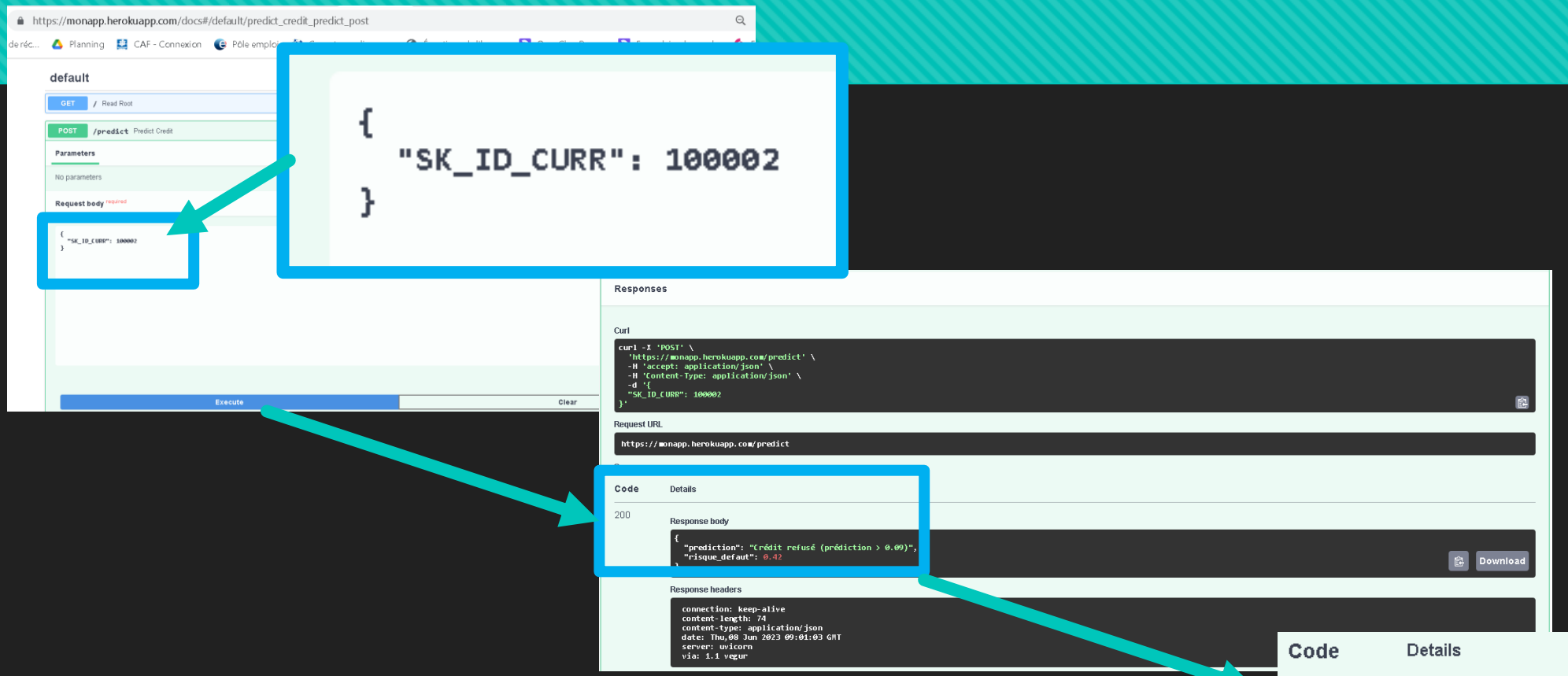


Recording experiments and models with MLFlow

- We record experiments in Microsof Azure studio :



API with FASTAPI and Dashboard with Streamlit



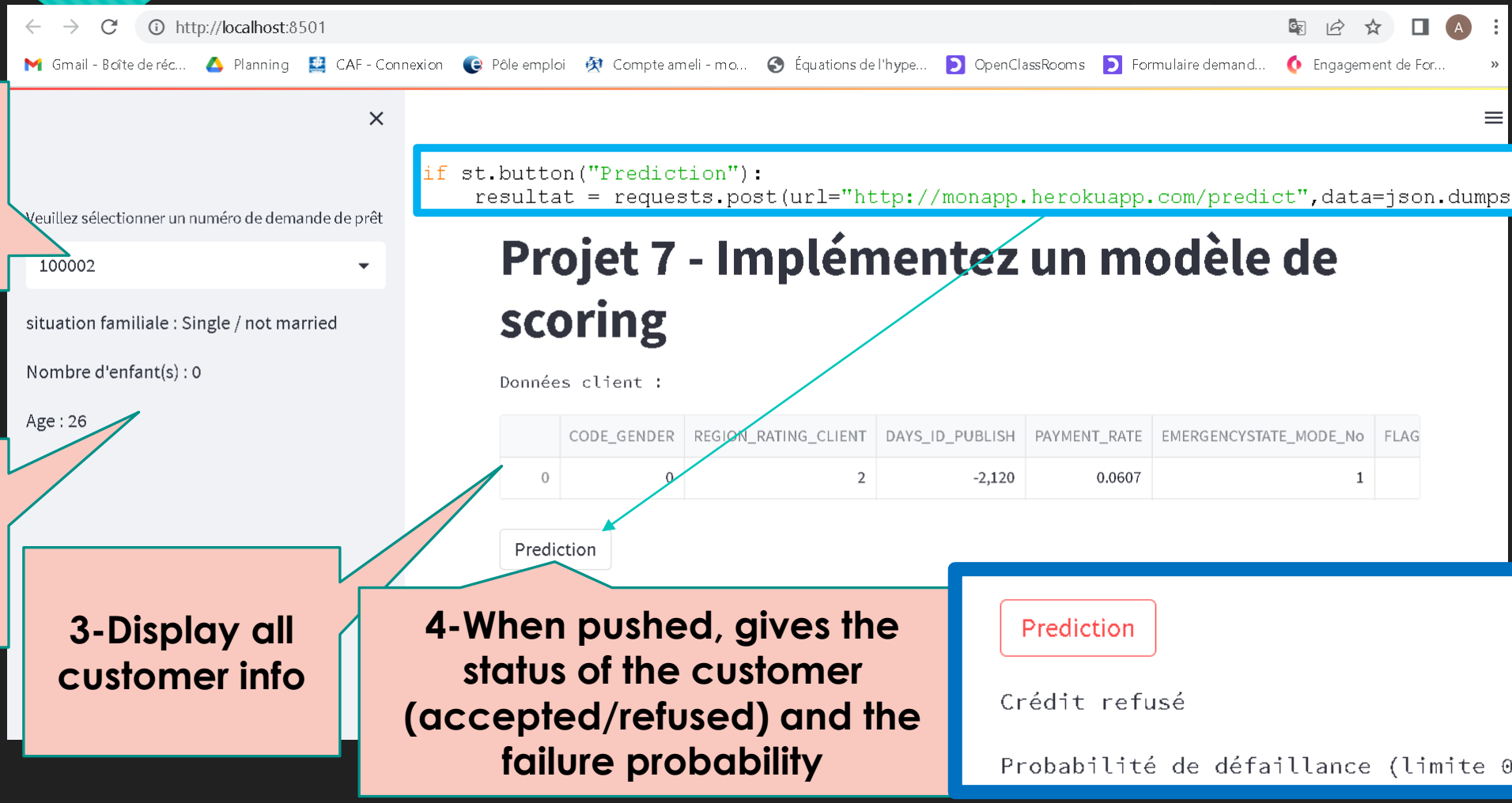
API with FASTAPI and Dashboard with Streamlit

1-Enter loan number

2-Display basic customer info

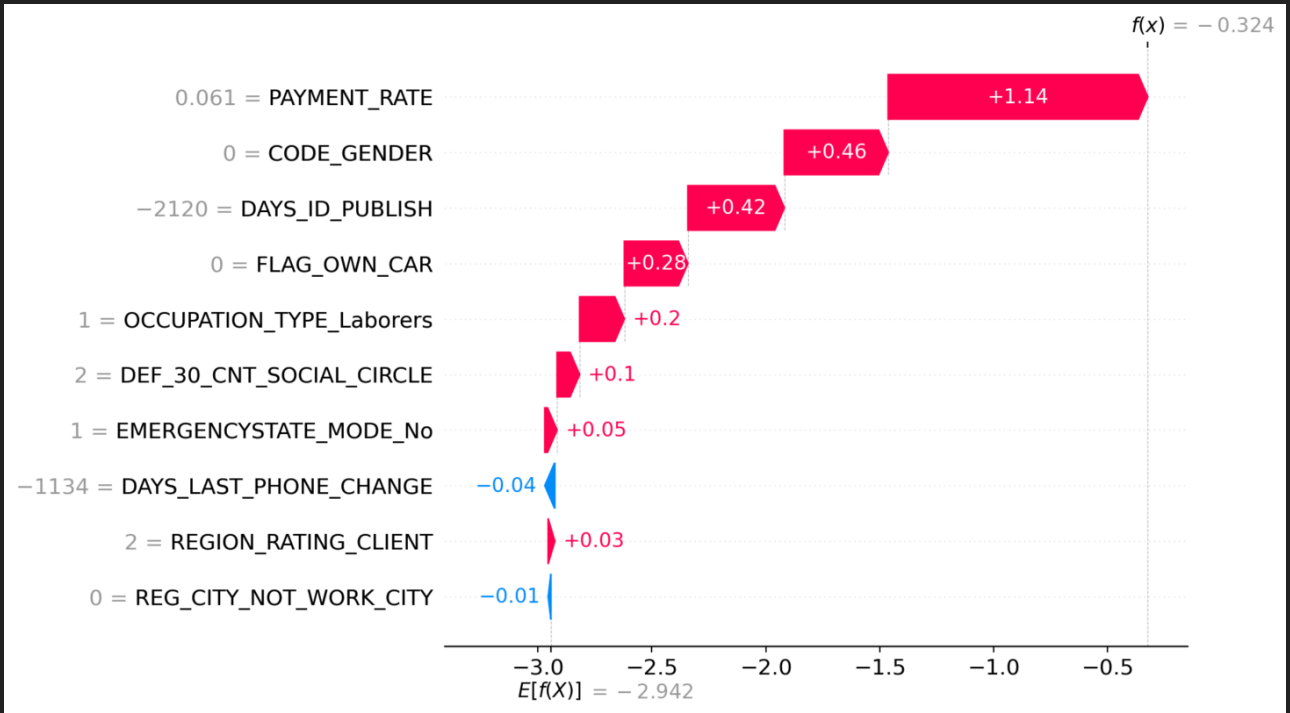
3-Display all customer info

4-When pushed, gives the status of the customer (accepted/refused) and the failure probability



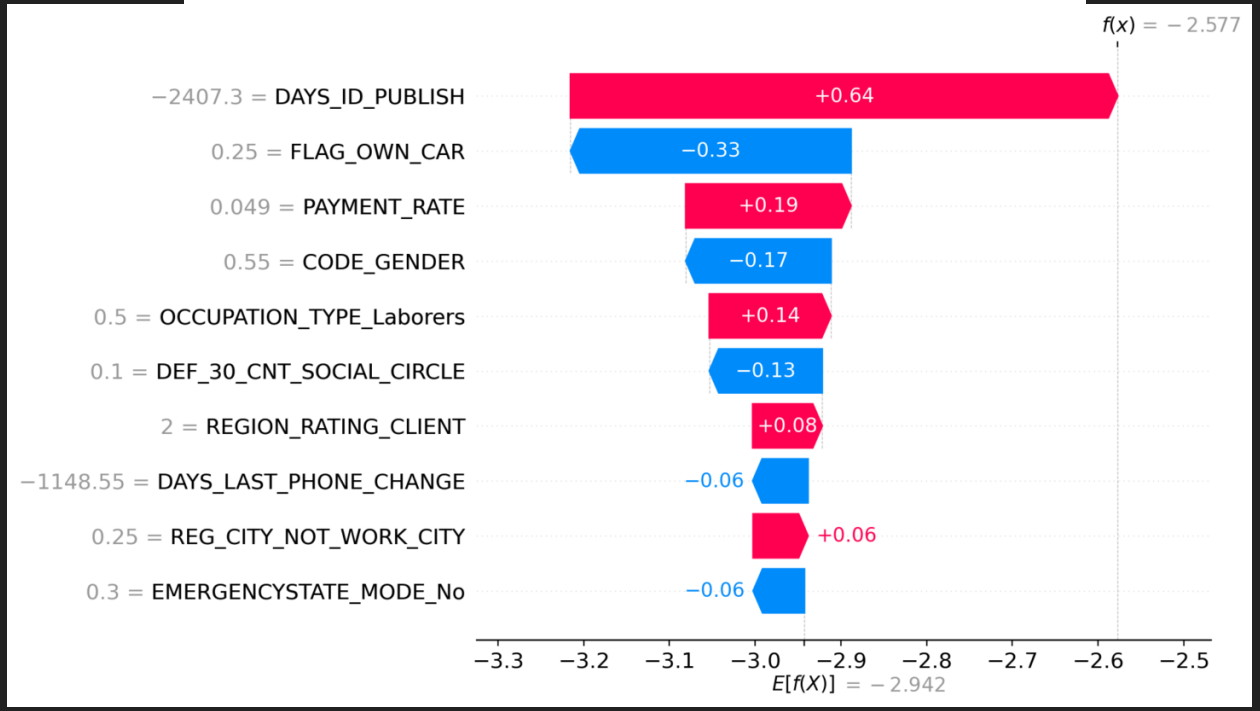
API with FASTAPI and Dashboard with Streamlit

Customer

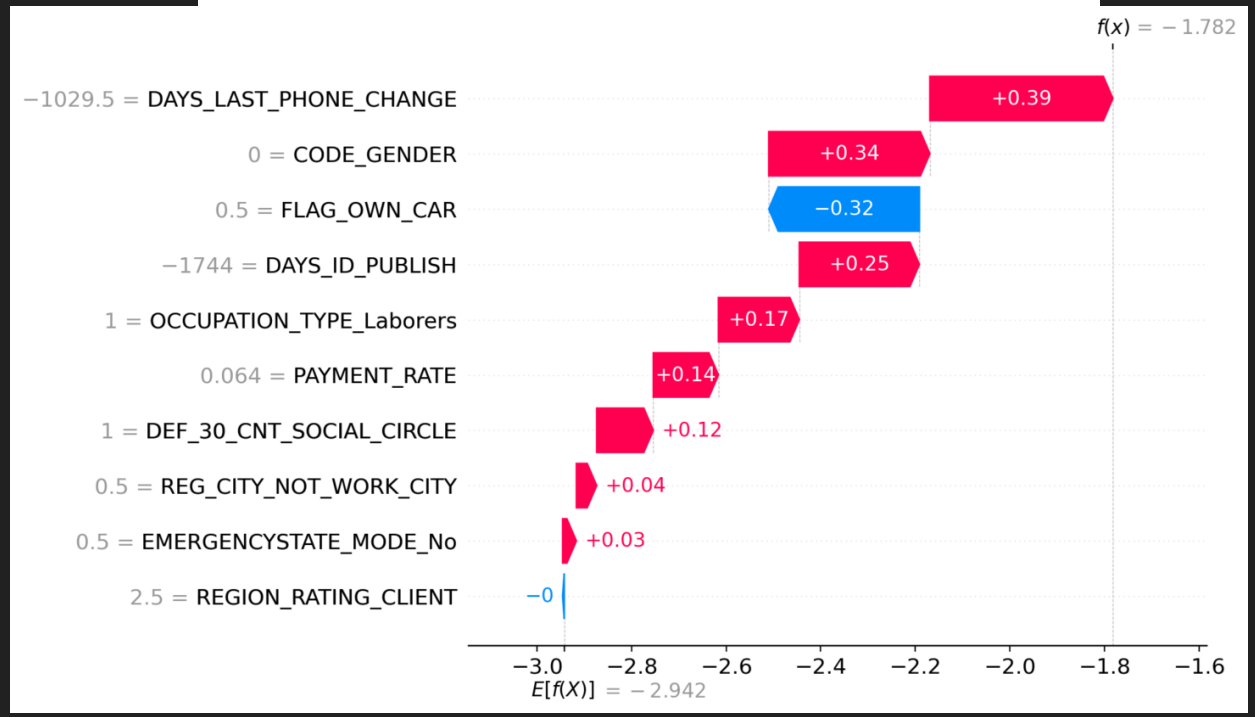


API with FASTAPI and Dashboard with Streamlit

Global population



Similar population (age, sex)



Testing with Pytest

- Two tests are implemented in the file « tests/test_P7.py » checking the ability of the algorithm to correctly predict an « accepted » and a « refused » customer.
- See the implementation of the tests below :

```
(venv) C:\Users\John\Desktop\Formation\7-Implémentez un modèle de scoring\tests>pytest
```

```
====== 2 passed, 6 warnings in 4.93s ==
```

Deployment of the API on internet with Heroku

- The tests are integrated in the API
  - In case of failure the user gets a message

Sharing code and versioning with Github

https://github.com/Antoine1608/OC-DS-P7

Gmail - Boîte de réc... Planning CAF - Connexion Pôle emploi Compte ameli - mo... Équations de l'hype... OpenClassRooms Form

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

Antoine1608 / OC-DS-P7 Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

About

Implémenter une application de scoring

Readme

Antoine1608 Update README.md 0b5ec53 now 24 commits

app	revue arboresence	last week
data	revue df limité aux features importantes	last week
model	new model and main improved on basic info	2 minutes ago
tests	new model and main improved on basic info	2 minutes ago
ui	new model and main improved on basic info	2 minutes ago
.gitignore	reinit gitignore	2 weeks ago
Naudy_Antoine_0_EDA_032023.ip...	mise à jour arborescence	last week
Naudy_Antoine_2_dossier_code_...	mise à jour arborescence	last week
Procfile	pytest	last week
README.md	Update README.md	now
note introductive.docx	inclusion des tests directement dans main.py	4 days ago
note méthodologique.docx	màj drif detection	last week
requirements.txt	arborescence refaite	2 weeks ago

README.md

Dashboard Streamlit de scoring déployé sur Heroku

Projet 7 du parcours Data Scientist

Un dashboard streamlit est déployé sur Heroku.

Il affiche des données et un graphe relatif à un client choisi :

https://github.com/Antoine1608/OC-DS-P7

Commit history

Commits

master

Commits on May 26, 2023

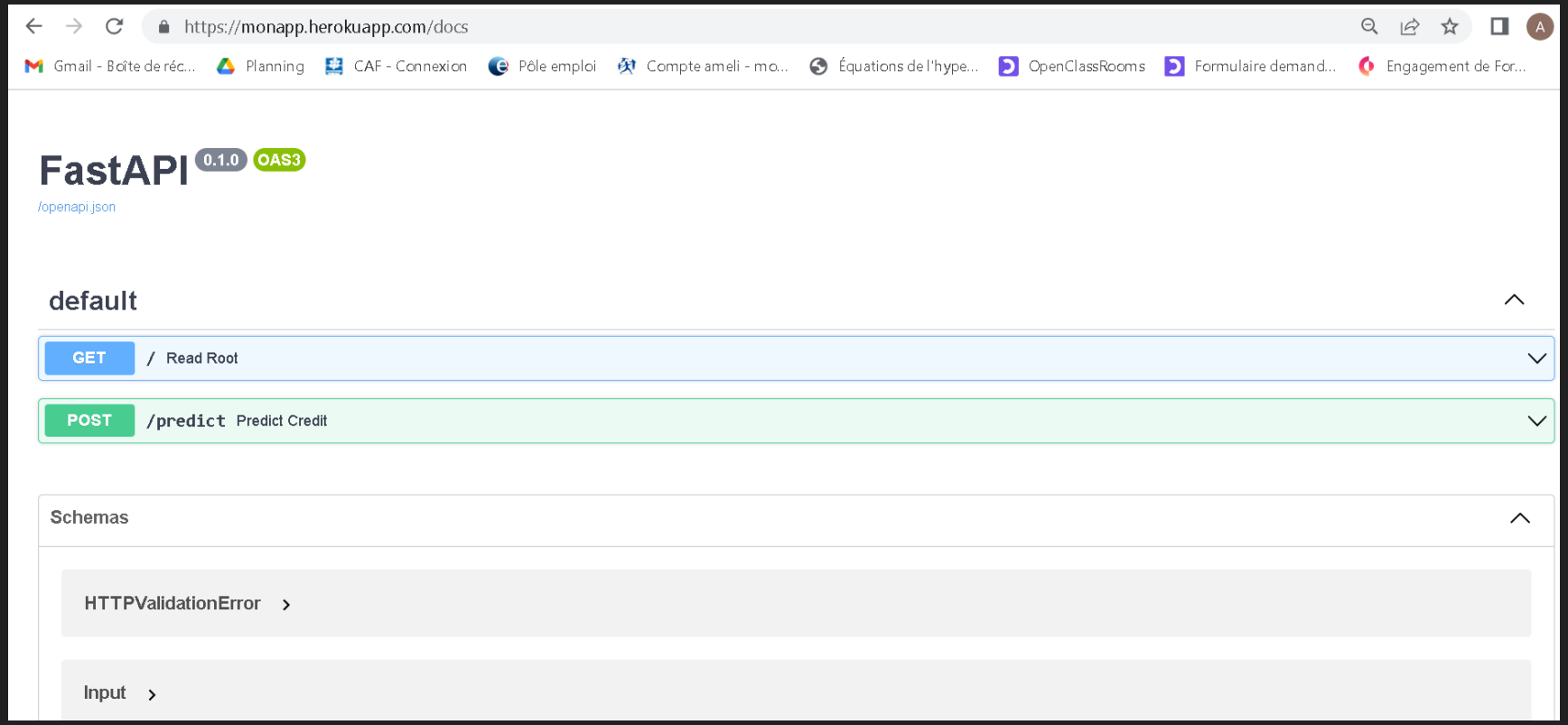
Update README.md	Antoine1608 committed 3 minutes ago	Verified 0b5ec53
new model and main improved on basic info	Antoine1608 committed 5 minutes ago	2e0efda

Commits on May 22, 2023

màj tests	Antoine1608 committed 4 days ago	2770f4e
testing tests	Antoine1608 committed 4 days ago	8cfe735
test de test	Antoine1608 committed 4 days ago	8db4dcf
supprimer 'import pytest'	Antoine1608 committed 4 days ago	9a29559
inclusion des tests directement dans main.py	Antoine1608 committed 4 days ago	ddcfc54

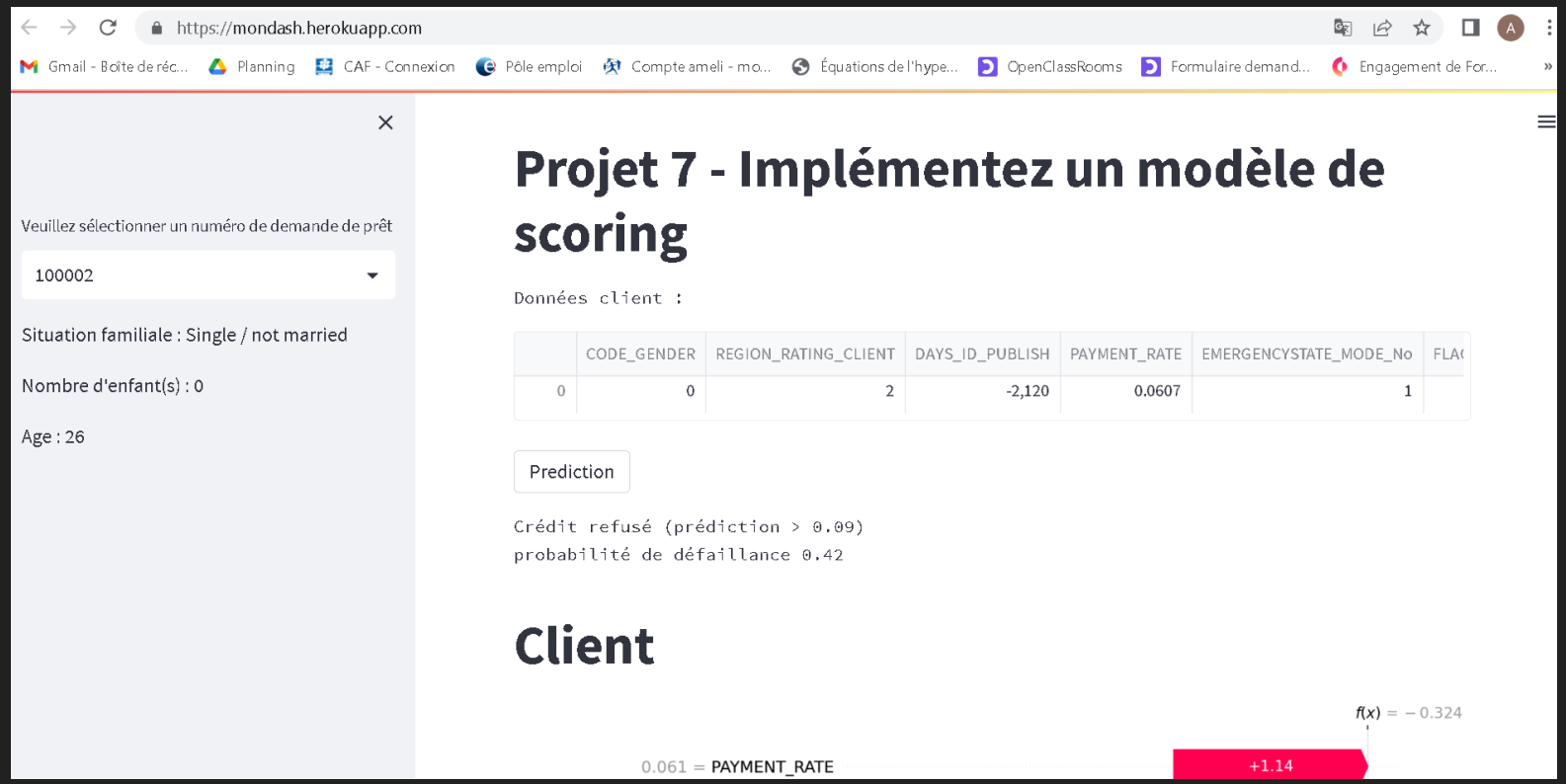
Deployment of the API on internet with Heroku

- The backend API can be accessed on the url :
  - https://monapp.herokuapp.com/docs



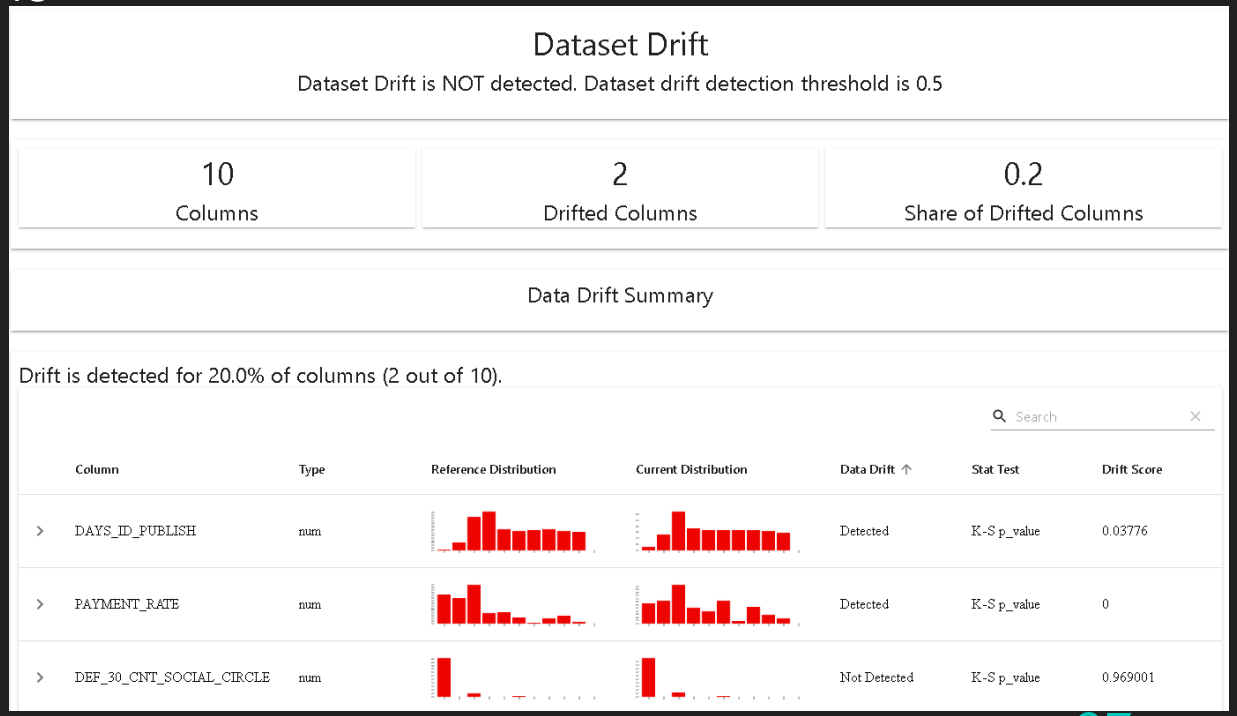
Deployment of the API on internet with Heroku

- The dashboard can be accessed on the url :
  - https://mondash.herokuapp.com



Drift analysis

- The drift analysis shows two features drifting between the « application_train » data and the « application_test » data :
  - DAYS_ID_PUBLISH with a K-S p_value of 0.03776
  - PAYMENT_RATE with a K-S p_value of 0
- See :
  - "tests\drift_report.html"



Dataset Drift

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

10	2	0.2
Columns	Drifted Columns	Share of Drifted Columns

Data Drift Summary

Drift is detected for 20.0% of columns (2 out of 10).

Search

	Column	Type	Reference Distribution	Current Distribution	Data Drift	Stat Test	Drift Score
>	DAYS_ID_PUBLISH	num			Detected	K-S p_value	0.03776
>	PAYMENT_RATE	num			Detected	K-S p_value	0
>	DEF_30_CNT_SOCIAL_CIRCLE	num			Not Detected	K-S p_value	0.969001

Conclusion

- FastAPI, Streamlit, Github and Heroku enable us to deploy swiftly an API with a dashboard on internet.
- By selecting 10 important features in the dataset we can show and explain easily the result of the prediction to the customer.
- However we have seen that data are drifting over time. It's necessary to re-train the model periodically to keep the good performance of the model.
- It's also possible, if time is available, to test other models with additional parameters to go further in the optimisation of the algorithms.