Introduction

Objectif du projet :

Je suis Data Scientist au sein d'une société financière, nommée **"Prêt à dépenser"**, qui propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d'historique de prêt.

L'entreprise souhaite **mettre en œuvre un outil de "scoring crédit" pour calculer la probabilité** qu'un client rembourse son crédit, puis classifie la demande en crédit accordé ou refusé. Elle souhaite donc développer un **algorithme de classification** en s'appuyant sur des sources de données variées (données comportementales, données provenant d'autres institutions financières, etc.).

De plus, les chargés de relation client ont fait remonter le fait que les clients sont de plus en plus demandeurs de **transparence** vis-à-vis des décisions d'octroi de crédit. Cette demande de transparence des clients va tout à fait dans le sens des valeurs que l'entreprise veut incarner.

Prêt à dépenser décide donc de **développer un dashboard interactif** pour que les chargés de relation client puissent à la fois expliquer de façon la plus transparente possible les décisions d'octroi de crédit, mais également permettre à leurs clients de disposer de leurs informations personnelles et de les explorer facilement.

Ce dashboard répond à des impératifs métiers, il doit être lisible, rapidement interprétable et doit pouvoir être montré au client. Les informations doivent donc être conformes à la RGPD et compréhensible aussi bien par un conseiller non informaticien et non data scientiste. La grande partie du travail sera simplifiée par la mise en place du modèle de machine Learning, mais le conseiller doit toujours être capable d'octroyer un crédit de son propre gré.

Afin de tenir compte de cette contrainte, les scores compris entre 45 % et 55 % seront affichés en orange, et à la main du conseiller.

Le découpage des dossiers est le suivant :

Architecture du repository et contenu des dossiers

Dossier api :

api
|----__pycache__
| api2.cpython-310.pyc
| __init.cpython-310.pyc
|---- static
| logo.jpg
|----templates
| index.html
|----test
| |-----__pycache
| |
| __init__.py
| test_api.py
api2.py
|
model_pipeline_with_params.joblib
|
requirements.txt
|
df_wk.csv
|
__init__.py
|
runtime.txt
|
Procfile

Contient les dossiers **__pycache__** , **static** qui contient les fichiers images du dashboard, **templates** avec le fichier index.html, structure html de l'api

Les fichiers de codes pour l'api sont :
api2.py : code de l'api
Model_pipeline_with_params.joblib : modèle

requirements.txt liste les bibliothèques nécessaires au fonctionnement de l'api.
__init__.py est un fichier nécessaire pour que python considère les dossiers comme contenant des paquets
runtime.txt est le fichier indiquant la version de python à Heroku
Procfile indique à heroku les commandes à effectuer pour démarrer le projet

/test/test_api.py : fichiers de test pytest

Dossier appli :

appli
|----__pycache__
| dash_api.cpython-310.pyc
| test_app.cpython-310 -7.4.0.pyc
|---- static
| logo.jpg
|----templates
| index.html
|----test
| |-----__pycache
| |
| __init_.py
| test_app.py
|
requirements.txt
|
runtime.txt
|
Procfile

Contient les dossiers **static** qui contient les fichiers images du dashboard, **templates** avec le fichier index.html, structure html de l'api

Les fichiers de codes pour l'api sont :
dash_api.py : code du dashboard
requirements.txt liste les bibliothèques nécessaires au fonctionnement de l'api.
runtime.txt est le fichier indiquant la version de python à Heroku
Procfile indique à heroku les commandes à effectuer pour démarrer le projet

/test/test_api.py : fichiers de test pytest