

The background is a dark blue gradient with a subtle pattern of white dots. On the left side, there are several concentric circles and a large circular scale with degree markings from 140 to 260. Some of the circles have arrows indicating a clockwise direction. The main title is centered on the right side in a large, bold, white sans-serif font.

IMPLÉMENTEZ UN MODÈLE DE SCORING

HAMIDI MOHAMED

PROBLEMATIQUE

- Développer un système de scoring crédit pour évaluer la probabilité de remboursement des clients
- Assurer une transparence via un dashboard interactif pour expliquer les décisions d'octroi de crédit.

Importance :

1. **Minimiser Risque Financier** : Réduction des pertes dues aux défauts de paiement.
2. **Confiance Client** : Transparence augmentant la satisfaction et fidélisation.
3. **Avantage Compétitif** : Innovation et transparence comme différentiateurs sur le marché.

JEU DE DONNÉES

- **application_train.csv / application_test.csv** : Données principales pour l'entraînement (avec TARGET) et le test (sans TARGET)
- **bureau.csv** : Historique des crédits des clients auprès d'autres institutions financières.
- **bureau_balance.csv** : Solde mensuel des crédits précédents.
- **POS_CASH_balance.csv** : Solde mensuel des précédents prêts et crédits consommation chez Home Credit.
- **credit_card_balance.csv** : Solde mensuel des cartes de crédit précédentes chez Home Credit.
- **previous_application.csv** : Toutes les précédentes demandes de prêts chez Home Credit.
- **installments_payments.csv** : Historique de remboursement des crédits précédemment accordés chez Home Credit.
- **HomeCredit_columns_description.csv** : Descriptions des colonnes des divers fichiers de données.

DÉMARCHE DE MODÉLISATION

- **Compréhension des Données** : Exploration des données pour identifier les tendances, anomalies et relations entre les variables.
- **Prétraitement des Données** :
 - Nettoyage.
 - Encodage des variables catégorielles.
 - Normalisation/Standardisation des variables numériques.
- **Features Engineering** : Création de nouvelles features
- **Partitionnement des Données**
- **Entraînement des Modèles** : Utilisation de différents algorithmes de classification (ex : Random Forest, Gradient Boosting, SVM).
- **Validation Croisée** : Utilisation de techniques de validation croisée pour garantir la robustesse du modèle.

MESURES D'ÉVALUATION

- **Score ROC AUC :**
 - Courbe ROC
 - AUC : Aire sous la courbe ROC
- **Coût Métier :**
 - Considération du coût métier déséquilibré entre un Faux Négatif (FN) et un Faux Positif (FP).
 - FN : mauvais client prédit comme bon client, entraînant une perte en capital.
 - FP : bon client prédit comme mauvais, entraînant un manque à gagner.
 - Supposition : le coût d'un FN est dix fois supérieur au coût d'un FP.
 - Élaboration d'un score "métier" pour minimiser le coût d'erreur de prédiction (FN et FP).

GESTION DU DÉSÉQUILIBRE DES CLASSES

- **Suréchantillonnage (Oversampling) :**
 - Augmentation de la taille de la classe minoritaire en dupliquant des échantillons.
- **Sous-échantillonnage (Undersampling) :**
 - Réduction de la taille de la classe majoritaire en supprimant des échantillons.
- **SMOTE (Synthetic Minority Over-sampling Technique) :**
 - Génération d'exemples synthétiques de la classe minoritaire pour améliorer l'équilibre des classes.

VISUALISATION DU TRACKING VIA MLFLOW

```
import mlflow

mlflow.set_experiment("Model_Selection")
mlflow.sklearn.autolog()
for model_name, model in classifieurs.items():
    if model_name in ['Dummy Classifier', 'Logistic Regression', 'Random Forest', 'K-Nearest Neighbors', 'Naive Bayes']:
        with mlflow.start_run(run_name = type(model).__name__):
            model.fit(X,y)

mlflow.lightgbm.autolog()
model = lgb.LGBMClassifier()
with mlflow.start_run(run_name = type(model).__name__):
    model.fit(X,y)

mlflow.xgboost.autolog()
model = XGBClassifier()
with mlflow.start_run(run_name = type(model).__name__):
    model.fit(X,y)
```

```
from pyngrok import ngrok
# Terminate open tunnels if exist
ngrok.kill()

# Setting the authtoken (optional)
# Get your authtoken from https://dashboard.ngrok.com/auth
NGROK_AUTH_TOKEN = "2VRaoEcG5R53V5nLrykMbBcM8yq_oUW1bYfhV1ReQkzdLozF"
ngrok.set_auth_token(NGROK_AUTH_TOKEN)

# Open an HTTPS tunnel on port 5000 for http://localhost:5000
ngrok_tunnel = ngrok.connect(addr="5000", proto="http", bind_tls=True)
print("MLflow Tracking UI:", ngrok_tunnel.public_url)

WARNING:pyngrok.process.ngrok:t=2023-09-23T23:04:14+0000 lvl=warn msg="ngrok config file found at legacy location, move to XDG location" xdg_path=/root/.config/ngrok/ngrok.yml legacy_path=/root/.ngrok2/ngrok.yml
MLflow Tracking UI: https://8c3c-34-83-285-143.ngrok-free.app
```

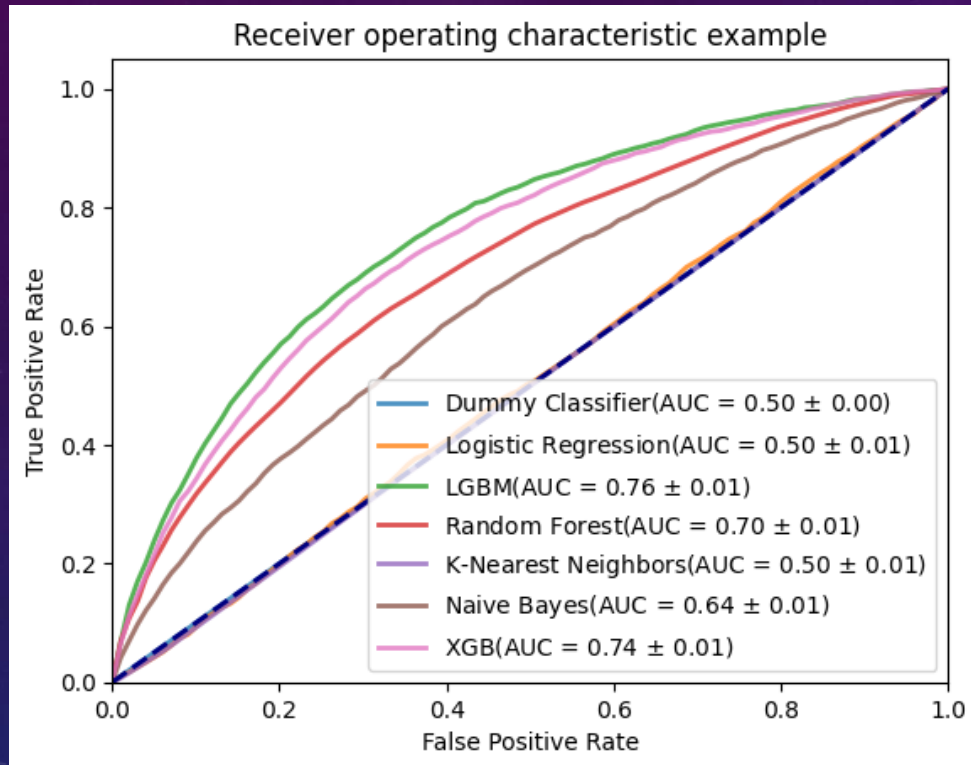
```
!mlflow ui

[2023-09-23 23:04:18 +0000] [71873] [INFO] Starting gunicorn 21.2.0
[2023-09-23 23:04:18 +0000] [71873] [INFO] Listening at: http://127.0.0.1:5000 (71873)
[2023-09-23 23:04:18 +0000] [71873] [INFO] Using worker: sync
[2023-09-23 23:04:18 +0000] [71874] [INFO] Booting worker with pid: 71874
[2023-09-23 23:04:18 +0000] [71875] [INFO] Booting worker with pid: 71875
[2023-09-23 23:04:18 +0000] [71876] [INFO] Booting worker with pid: 71876
[2023-09-23 23:04:18 +0000] [71877] [INFO] Booting worker with pid: 71877
```

RESULTAT DU TRACKING

Table Chart Evaluation Experimental										
				Metrics						
<input type="checkbox"/>		Run Name	Created	Duration	training_accurac	training_f1_score	training_log_loss	training_precision	training_recall_score	training_roc_auc
<input type="checkbox"/>		XGBClassifier	33 minutes ago	1.6min	-	-	-	-	-	-
<input type="checkbox"/>		LGBMClassifier	34 minutes ago	18.9s	-	-	-	-	-	-
<input type="checkbox"/>		GaussianNB	34 minutes ago	8.3s	0.92	0.881	0.278	0.846	0.92	0.622
<input type="checkbox"/>		KNeighborsClassifier	39 minutes ago	5.3min	0.921	0.887	0.181	0.898	0.921	0.88
<input type="checkbox"/>		RandomForestClassifier	40 minutes ago	1.0min	1	1	0.061	1	1	1
<input type="checkbox"/>		LogisticRegression	40 minutes ago	8.5s	0.92	0.881	0.309	0.846	0.92	0.509
<input type="checkbox"/>		DummyClassifier	40 minutes ago	13.8s	0.92	0.881	0.279	0.846	0.92	0.5

SYNTHÈSE DES RÉSULTATS



- Score ROC-AUC : 0,67
- Score métier : 0,68

PRÉSENTATION DE LA PIPELINE DE DEPLOIEMENT

Nom	Modifié le	Type	Taille
.git	21/09/2023 21:25	Dossier de fichiers	
__pycache__	22/09/2023 00:49	Dossier de fichiers	
app	21/09/2023 21:24	Python File	2 Ko
Procfile	21/09/2023 21:08	Fichier	1 Ko
requirements	21/09/2023 20:15	Document texte	1 Ko

Nom	Modifié le	Type	Taille
.git	23/09/2023 14:37	Dossier de fichiers	
dashboard	21/09/2023 21:28	Python File	26 Ko
Procfile	21/09/2023 18:35	Fichier	1 Ko
requirements	21/09/2023 18:30	Document texte	1 Ko
setup	21/09/2023 18:34	Shell Script	1 Ko

MohamedData / Projet_7

Type [] to search

> + 🔍 📄 📧 👤

<> Code 🔔 Issues 📄 Pull requests ⌚ Actions 📁 Projects 📖 Wiki 🔒 Security 📈 Insights ⚙️ Settings

👤 **Projet_7** Public

📌 Pin ⌚ Unwatch 1 🍴 Fork 0 ⭐ Star 0

📁 main 1 branch 0 tags

Go to file Add file <> Code

👤 MohamedData Ajout des fichiers pickle 38324ee 7 hours ago 2 commits

📄 best_model.pickle Ajout des fichiers pickle 7 hours ago

📄 test_prediction.pickle Ajout des fichiers pickle 7 hours ago

📄 test_set.pickle Ajout des fichiers pickle 7 hours ago

Help people interested in this repository understand your project by adding a README.

Add a README

About

No description, website, or topics provided.

🔔 Activity

⭐ 0 stars

👁 1 watching

🍴 0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

COMMITTS

Invite de commandes

```
C:\Users\nessi\deploiement_dashboard>cd C:\Users\nessi\
```

```
C:\Users\nessi>cd p_7
```

```
C:\Users\nessi\P_7>cd api
```

```
C:\Users\nessi\P_7\api>git log
```

```
commit d0934bf2d987a8d6d09eaa5a0a0fa69b4a541517 (HEAD -> master, heroku/master)
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 21:25:29 2023 +0200
```

suppression du setup et test avec chatgpt

```
commit b1297052eb518cf78400ff1eb24d824d68fc9798
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 21:10:36 2023 +0200
```

Nouvel essai de déploiement Heroku

```
commit 41293c54474b0df1caa0aedf90029959b6596cb1
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 20:16:35 2023 +0200
```

troisieme essai en modifiant requirements une nouvelle fois

```
commit 83858385d991ef0c00e4953cb63143da24446d5
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 20:12:10 2023 +0200
```

deuxieme essai en mettant toutes les librairies

```
commit 980480e369498039d434dcb3dd8b2d4cfb7c2931
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 19:58:53 2023 +0200
```

premier essai

Invite de commandes

```
C:\Users\nessi>cd str_env
```

```
C:\Users\nessi\str_env>cd Scripts
```

```
C:\Users\nessi\str_env\Scripts>cd activate
```

Nom de répertoire non valide.

```
C:\Users\nessi\str_env\Scripts>cd C:\Users\nessi\
```

```
C:\Users\nessi>cd deploiement_dashboard
```

```
C:\Users\nessi\deploiement_dashboard>git log
```

```
commit 27e2bf73e48743082e676fe2ded18f290733b911 (HEAD -> master, heroku/master)
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 21:30:44 2023 +0200
```

mise a jour lien api

```
commit 5f68df1ef2101cde7f5473cab77c24da016ff1c2
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 19:29:58 2023 +0200
```

modification dashboard pour ne plus avoir erreur sur dash_loc

```
commit 5ef81c64012aff498677fa7aaa3e367babf45e01
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 19:19:35 2023 +0200
```

troisieme essai en retirant les points de procfile et setup.sh

```
commit da214b83a62efd9e9fa864a2162b47070359381c
```

```
Author: Mohamed Hamidi <mohamedhpro@outlook.fr>
```

```
Date: Thu Sep 21 19:03:02 2023 +0200
```

deuxieme essai

TESTS UNITAIRES (PYTEST)

```
# tests/test_app.py
from api.app import app
import json
from api.app import predict_proba
import numpy as np

client = app.test_client()

def test_predict_proba_route():
    response = client.get('/predict_proba/208550')
    data = json.loads(response.data)
    assert 'proba_classe_0' in data
    assert 'proba_classe_1' in data

def test_predict_proba_function():
    proba = predict_proba(208550)
    assert isinstance(proba, np.ndarray)
    assert 0 <= proba[0] <= 1
    assert 0 <= proba[1] <= 1
```

```
(str_env) C:\Users\nessi\P_7>pytest
```

```
===== test session starts =====
platform win32 -- Python 3.9.17, pytest-7.4.2, pluggy-1.3.0
rootdir: C:\Users\nessi\P_7
plugins: flask-1.2.0
collected 2 items
```

```
tests\test_app.py ..
```










```
[100%]
```

```
===== 2 passed in 4.94s =====
```

```
(str_env) C:\Users\nessi\P_7>
```


ANALYSE DU DATADrift (EVIDENTLY)

Le Data Drift (ou dérive des données) est un phénomène qui se produit lorsque les données sur lesquelles un modèle a été entraîné (données d'entraînement) commencent à différer des données que le modèle rencontre en production (données en direct).

Drift is detected for 7.44% of features (9 out of 121). Dataset Drift is NOT detected.						
<div>Q Search X</div>						
Feature	Type	Reference Distribution	Current Distribution	Data Drift	Stat Test	Drift Score
> AMT_REQ_CREDIT_BUREAU_QRT	num			Detected	Wasserstein distance (normed)	0.359052
> AMT_REQ_CREDIT_BUREAU_MON	num			Detected	Wasserstein distance (normed)	0.281765
> AMT_GOODS_PRICE	num			Detected	Wasserstein distance (normed)	0.210785
> AMT_CREDIT	num			Detected	Wasserstein distance (normed)	0.207334
> AMT_ANNUITY	num			Detected	Wasserstein distance (normed)	0.161102
> AMT_REQ_CREDIT_BUREAU_WEEK	num			Detected	Wasserstein distance (normed)	0.15426
> NAME_CONTRACT_TYPE	cat			Detected	Jensen-Shannon distance	0.14755
> DAYS_LAST_PHONE_CHANGE	num			Detected	Wasserstein distance (normed)	0.138977
> FLAG_EMAIL	num			Detected	Jensen-Shannon distance	0.122121

API (FLASK)

```
# Importation des librairies
from flask import Flask, request, jsonify
import pickle
import pandas as pd
from lightgbm import LGBMClassifier
import os

#Création d'une instance de Flask
app = Flask(__name__)


# Chargement du dataset de test
dataset = pd.read_pickle('https://github.com/MohamedData/Projet_7/raw/main/test_set.pickle')

# Chargement du modèle
model = pd.read_pickle('https://github.com/MohamedData/Projet_7/raw/main/best_model.pickle')

# Fonction de prédiction
def predict_proba(index):
    data = dataset.loc[index]
    proba = model.predict_proba([data])[0]
    return proba

# Récupération de l'index via l'URL
@app.route('/predict_proba/<int:index>', methods=['GET'])
# Fonction qui à un index renvoie les probabilités d'appartenance aux classes 0 et 1
def get_prediction_proba(index):
    proba = predict_proba(index)
    return jsonify({'proba_classe_0': float(proba[0]), 'proba_classe_1': float(proba[1])})

# Démarrage de l'application
if __name__ == '__main__':
    port = int(os.environ.get("PORT", 5000))
    app.run(host='0.0.0.0', port=port)
```



← → ↻ api-hm-8e0d0e66dd33.herokuapp.com/predict_proba/208550

Musique OpenClassrooms Machine Learning GitHub Montage G

{"proba_classe_0":0.9999396247278981,"proba_classe_1":6.0375272101944164e-05}

DASHBOARD (STREAMLIT)

```
def main():  
    st.set_page_config(page_title="Mon Dashboard")  
  
    #st.title("Mon Dashboard")  
    afficher_page_accueil()  
    # Créer un menu de navigation à gauche  
    selection = st.selectbox("Sélectionnez une option", ["Sélectionnez une option", "Global", "Local"])  
    if selection == "Global":  
        lancer_dash_glo()  
    elif selection == "Local":  
        lancer_dash_loc()
```

- Page d'accueil
- Dashboard Global :
 - Analyse d'ensemble des features importantes
 - Mise en relation des valeurs du clients avec celles d'un échantillon
- Dashboard Local : Analyse locale des variables déterminantes pour l'accès (ou non) au crédit

DÉMONSTRATION



CONCLUSION

- Modélisation : EDA, Pré-traitement, features engineering, modélisation, optimisation (SMOTE)
- Utilisation de MIFlow pour le tracking
- Utilisation de Evidently pour le Data Drift : Pas de présence de data drift
- Utilisation de Flask pour monter une API
- Utilisation de Streamlit pour monter un Dashboard
- Déploiement via Git, GitHub et Heroku