Implémentez un modèle de scoring

Yihan ZHANG

Implémentez un modèle de scoring

01	**“Prêt à dépenser”**	• société financière qui propose des crédits à la consommation pour des personnes ayant peu ou pas du tout d’historique de prêt.
02	**Projet**	• Un outil de “scoring crédit” pour calculer la probabilité qu’un client rembourse son crédit, puis classifie la demande en crédit accordé ou refusé. • Un dashboard interactif pour que les chargés de relation client puissent expliquer de façon transparente les décisions d’octroi de crédit, et que leurs clients puissent disposer de leurs informations personnelles.
03	**Notre mission**	• Construire un modèle de scoring qui donnera une prédiction sur la probabilité de remboursement d'un client. • Construire un dashboard interactif permettant aux gestionnaires d'interpréter les prédictions faites par le modèle. • Mettre en production le modèle de prédiction à l’aide d’une API, ainsi que le dashboard qui appelle l’API pour les prédictions.

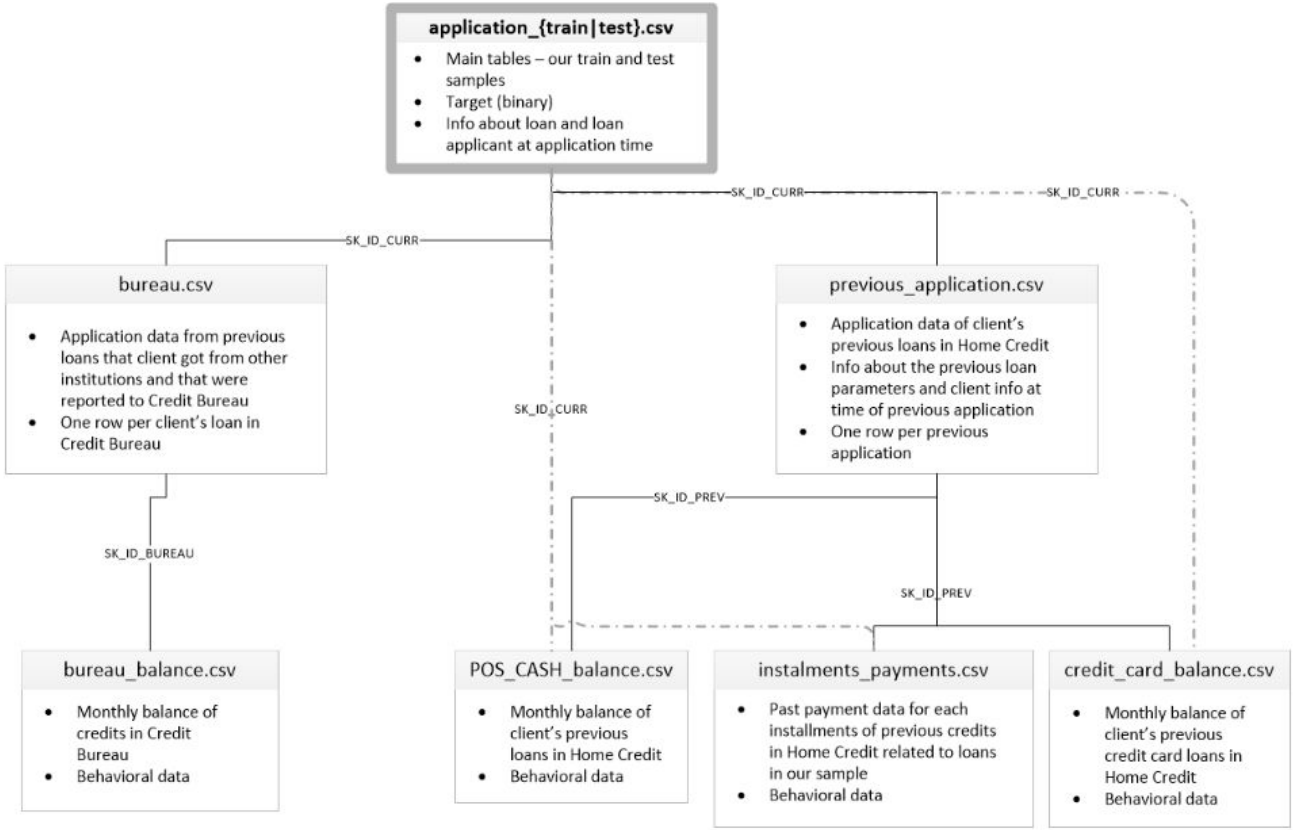
Implémentez un modèle de scoring

- Environnement technique
- Exploration du jeu de données
- Feature engineering
- Préparation des données
- Modélisation
- Démarche MLOps
- API
- Dashboard
- Déploiement continu de l'API et du dashboard
- Analyse du data drift
- Conclusion

Environnement technique

- Jupyter Notebook - Google Colab
- Python 3.10
  - Pandas, Numpy
  - Matplotlib, Seaborn
  - Scikit-Learn
  - MLFlow
  - Evidently AI
- Flask API
- Github
- Streamlit
- Heroku

Exploration du jeu de données



Le dataset contient plusieurs fichiers relatifs aux prêts des clients.

application_train/test.csv (données de prêt)

bureau.csv et bureau_balance.csv (crédits antérieurs signalés au bureau de crédit)

POS_CASH_balance.csv et credit_card_balance.csv (soldes mensuels des crédits à la consommation et des cartes de crédit)

previous_application.csv (demandes de prêts précédentes)

installments_payments.csv (historique des remboursements).

Un fichier de description des colonnes est également inclus.

Feature engineering

Encodage des variables catégorielles	Prétraitement des fichiers de données	Création de nouvelles features	Agrégation et jonction des données
Les colonnes catégorielles sont transformées en variables binaires (one-hot encoding) à l'aide de la fonction one_hot_encoder.	Chaque type de données (crédits antérieurs, demandes précédentes, paiements d'échéances, etc.) est agrégé par client pour être joint à la table principale.	Des nouvelles features sont créées à partir des features importantes, comme le taux de paiement (`PAYMENT_RATE`), etc.	Toutes les tables sont agrégées par l'identifiant client (`SK_ID_CURR`) et jointes à la table principale `application_train_test`.

Préparation des données

1. Valeurs manquantes
2. Data splitting
3. Imputations
4. Équilibre des classes

Préparation des données

1 Valeurs manquantes

Supprimer les colonnes où le taux de remplissage est inférieur à 60% (à l'exception de la colonne "TARGET").

	nb_values	missing_values	filling_rate
ACTIVE_AMT_CREDIT_SUM_LIMIT_MEAN	218100.0	138151.0	0.612209
CLOSED_AMT_CREDIT_SUM_LIMIT_MEAN	229937.0	126314.0	0.645435
ACTIVE_AMT_CREDIT_SUM_DEBT_MEAN	239860.0	116391.0	0.673289
ACTIVE_AMT_CREDIT_SUM_DEBT_MAX	239860.0	116391.0	0.673289
ACTIVE_DAYS_CREDIT_ENDDATE_MIN	241342.0	114909.0	0.677449
ACTIVE_DAYS_CREDIT_ENDDATE_MAX	241342.0	114909.0	0.677449
ACTIVE_DAYS_CREDIT_ENDDATE_MEAN	241342.0	114909.0	0.677449
APPROVED_APP_CREDIT_PERC_VAR	249103.0	107148.0	0.699235
ACTIVE_AMT_CREDIT_SUM_MEAN	251808.0	104443.0	0.706827
ACTIVE_AMT_CREDIT_SUM_MAX	251808.0	104443.0	0.706827
ACTIVE_AMT_CREDIT_SUM_DEBT_SUM	251811.0	104440.0	0.706836

Préparation des données

2 Data splitting

Extraction de la colonne cible (TARGET)

Sélection des fonctionnalités (features)

Séparation des données d'entraînement (70%) et de validation (30%)

Préparation des données

3 Imputations

Réaliser l'imputation des valeurs manquantes dans les ensembles de données d'entraînement et de validation **séparément** en utilisant la classe `SimpleImputer` du module `impute` de scikit-learn.

- Éviter le **Data Leakage**

Taux de valeurs manquantes avant imputation - données d'entraînement: 0.0655

Taux de valeurs manquantes avant imputation - données de validation : 0.0635

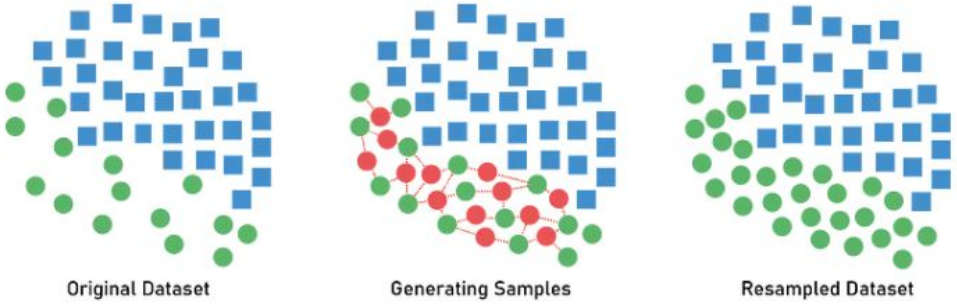
Taux de valeurs manquantes après imputation - données d'entraînement: 0.0

Taux de valeurs manquantes après imputation - données de validation: 0.0

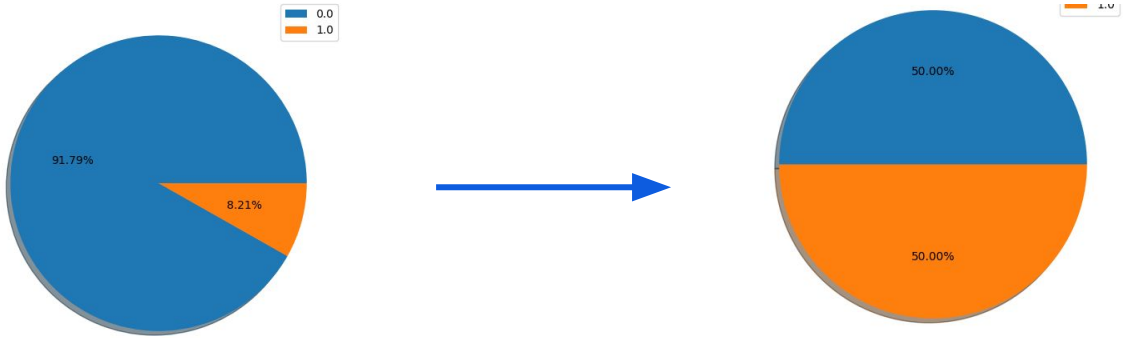
Préparation des données

4 Équilibre des classes

Synthetic Minority Oversampling Technique



SMOTE (Synthetic Minority Oversampling Technique): Génération d'échantillons pour la classe minoritaire en combinant les caractéristiques de ses voisins les plus proches.



Modélisation

- Objectif : prédire la probabilité de remboursement du crédit par le client, et classifier la demande en crédit “accordé” ou crédit ”refusé”.
- Solution: classification binaire (apprentissage supervisé)

1. Custom scoring function
2. Grid Search Cross Validation
3. Comparaison et choix du modèle
4. Entraînement & Prédiction

Modélisation

1 Custom scoring function

		Vraie valeur	
		- TARGET=0	+ TARGET=1
Valeur prédite	- TARGET =0	Client prédit sans difficulté de remboursement, et réellement sans difficulté de remboursement (TN)	⚠⚠⚠ Octroi de crédit à un client qui aura des difficultés de remboursement (FN) ⇒ Pertes en capital, frais.
	+ TARGET =1	Refus de crédit à un client qui n'aurait pas eu de difficulté de remboursement (FP) ⇒ Manque à gagner.	Client prédit avec difficultés de remboursement, et réellement avec difficultés de remboursement (TP)

Objectifs :

- Maximiser la capacité à détecter tous les clients avec difficultés de paiement, cad maximiser le rappel : TP / (TP+FN).
- Maximiser la capacité à ne pas détecter comme "mauvais" un "bon" client, cad maximiser la précision : TP / (TP+FP).

Définition d'une fonction de coût à optimiser:

- Hypothèse : le coût d'un FN est 10 fois supérieur au coût d'un FP
- score = 1*fp + 10*fn

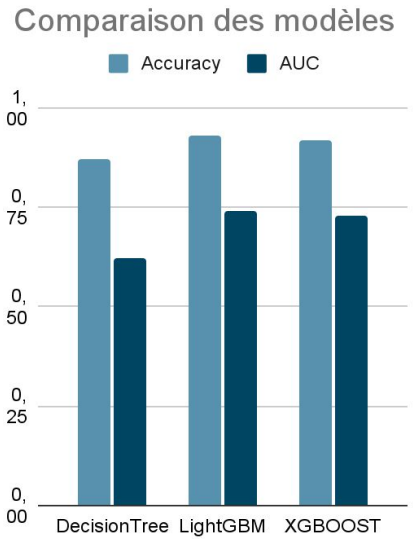
Modélisation

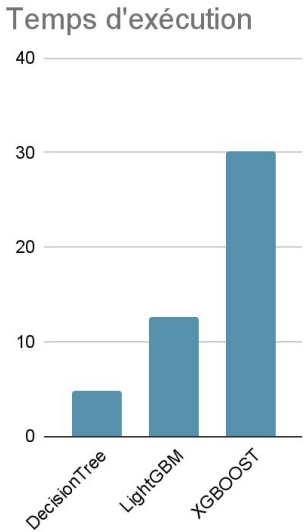
2 Grid Search Cross Validation

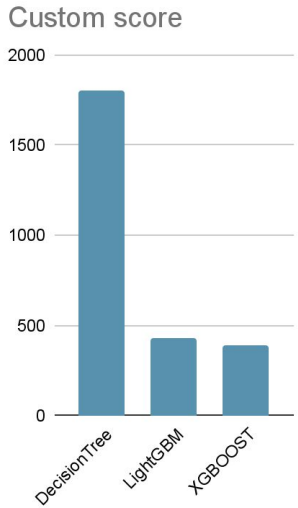
3 modèles : DecisionTree, LightGBM, XGBOOST

- Recherche des meilleurs hyperparamètres à l'aide de GridSearchCV
- Score d'optimisation : Custom scoring function.

3 Comparaison et choix du modèle







Meilleur Modèle : LightGBM

4 Entraînement & Prédiction

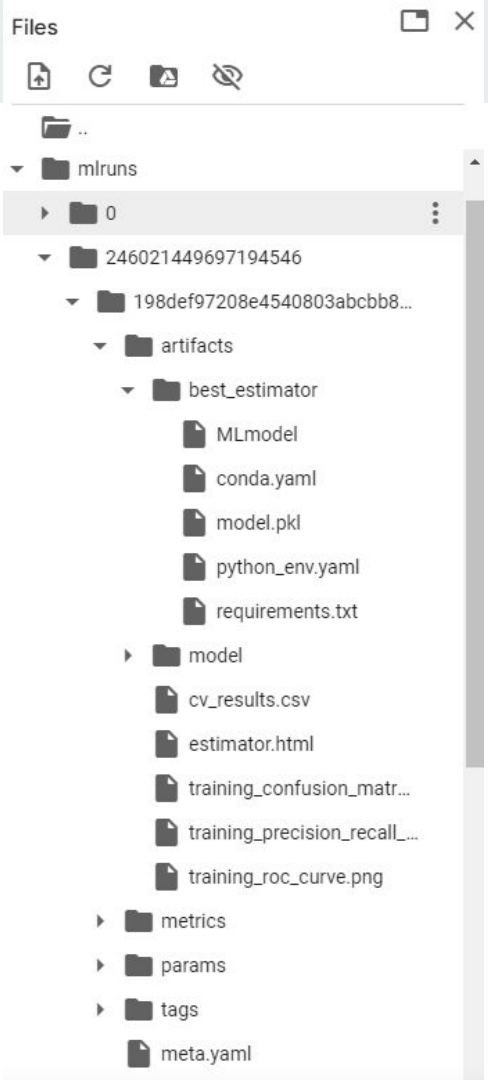
Prédiction sur les données de test :
- Probabilité de remboursement avec predict_proba()
- classe avec predict()

Démarche MLOps

Mettre en oeuvre d'une démarche de type MLOps d'automatisation et d'industrialisation de la gestion du cycle de vie du modèle.

- MLFlow pour la gestion "d'expériences" et leur tracking lors de la phase d'entraînement des modèles
- MLFlow pour le stockage centralisé des modèles dans un "model registry" et le serving
- Git, logiciel de version de code, pour suivre les modifications du code final de l'API de prédiction de tags à déployer
- Github pour stocker et partager sur le cloud le code de l'API, alimenté par un "push" Git et ainsi assurer une intégration continue

https://github.com/yihanzh/projectscoring



API

Mettre en production le modèle de scoring de prédiction à l'aide d'une API, ainsi que le dashboard interactif qui appelle l'API pour les prédictions.

- Flask est un micro-framework web en Python qui permet de créer des applications web de manière simple et rapide. Il est léger, flexible et facilement extensible grâce à de nombreuses extensions. Une des utilisations courantes de Flask est la création d'API (Application Programming Interface) pour permettre la communication entre différentes applications ou services.

https://github.com/yihanzh/projectscoring/blob/main/app.py

Endpoint	Description
/api/ping/ (GET)	Vérifie la disponibilité du service.
/api/index/ (GET)	Récupère les indices du DataFrame.
/api/features/ (GET)	Récupère les noms des colonnes (features) du DataFrame.
/api/data/ (GET)	Récupère les données d'une ligne spécifique du DataFrame en fonction de l'index fourni.
/api/predict/ (GET)	Effectue une prédiction en utilisant le modèle sur la ligne spécifiée par l'index.
/api/predict_proba/ (GET)	Calcule la probabilité de prédiction pour la ligne spécifiée par l'index.
/api/shap_global/ (GET)	Génère et renvoie une image du graphique SHAP global pour les principales features.
/api/shap_local/ (GET)	Génère et renvoie une image du graphique SHAP local pour une ligne spécifique et les principales features.
/api/distribution_feature/ (GET)	Génère et renvoie une image de la distribution d'une feature spécifique pour une ligne donnée.
/api/bivariate_plot/ (GET)	Génère et renvoie une image d'un graphique bivarié pour deux features spécifiques et une ligne donnée.

Dashboard

Spécifications du dashboard

- Permettre de visualiser le score et l'interprétation de ce score pour chaque client de façon intelligible pour une personne non experte en data science.
- Permettre de visualiser des informations descriptives relatives à un client (via un système de filtre).
- Permettre de comparer les informations descriptives relatives à un client à l'ensemble des clients ou à un groupe de clients similaires.

Solution technique : Streamlit

- Framework Python open-source qui permet de créer rapidement des tableaux de bord interactifs pour visualiser et partager des données.
- Utilisé pour le développement de la partie interface utilisateur.
- Interroge l'API de prédiction pour obtenir les informations demandées par l'utilisateur.

https://testyihanscoring1-94660854c600.herokuapp.com/

Prêt à dépenser

Sélectionner l'index du client :

1

	1
CODE_GENDER	1
FLAG_OWN_CAR	0
FLAG_OWN_REALTY	0
CNT_CHILDREN	2
AMT_INCOME_TOTAL	112500.0
AMT_CREDIT	90000.0
AMT_ANNUITY	9580.5
AMT_GOODS_PRICE	90000.0
REGION_POPULATION_RELATIVE	0.04622
DAYS_BIRTH	-9764

Tableau de bord - "Prêt à dépenser"

Prédiction

Client index: 1

- Probabilité de rembousement du client sélectioné: **90.00** %
- La réponse suggérée pour la demande de prêt du client sélectioné est :

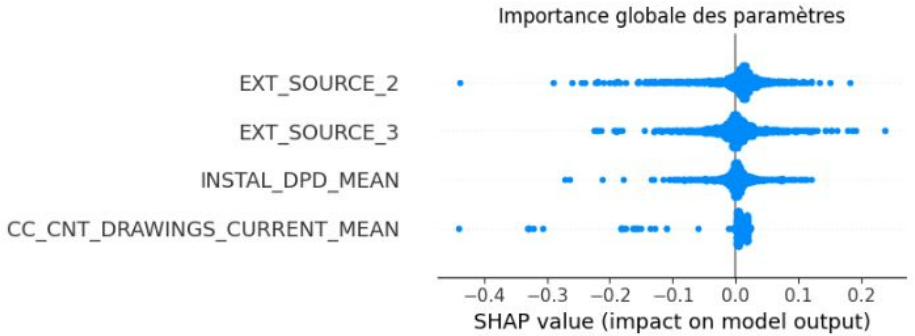
Crédit accordé !

Sélectionner le nombre de paramètres pour expliquer la prédiction

4

2 30

Importance globale des paramètres



Déploiement continu de l'API et du dashboard

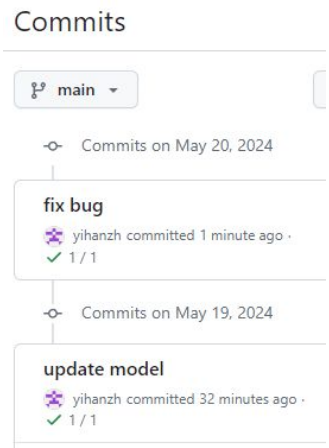
commit & push

Github Actions
Tests unitaires

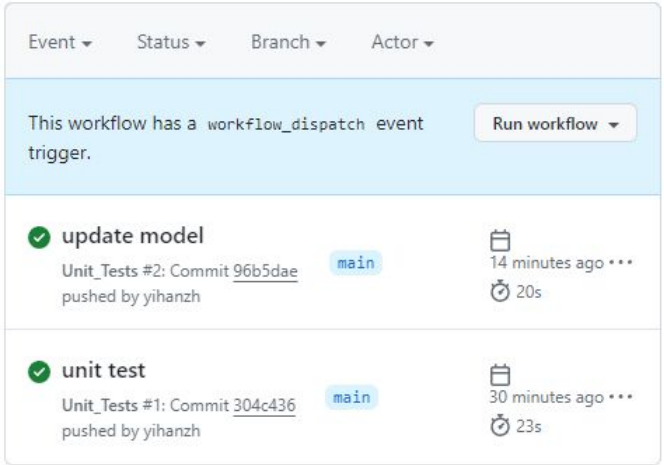
Déploiements
automatiques déclenchés
par Heroku

Nouveau
dashboard
disponible

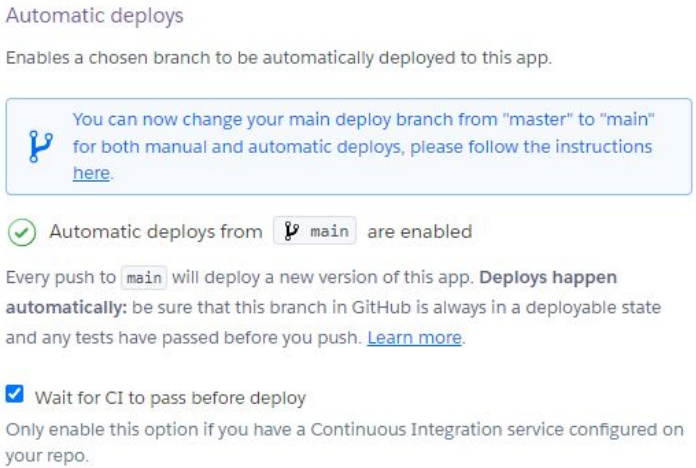
Local Git directory



Github repository



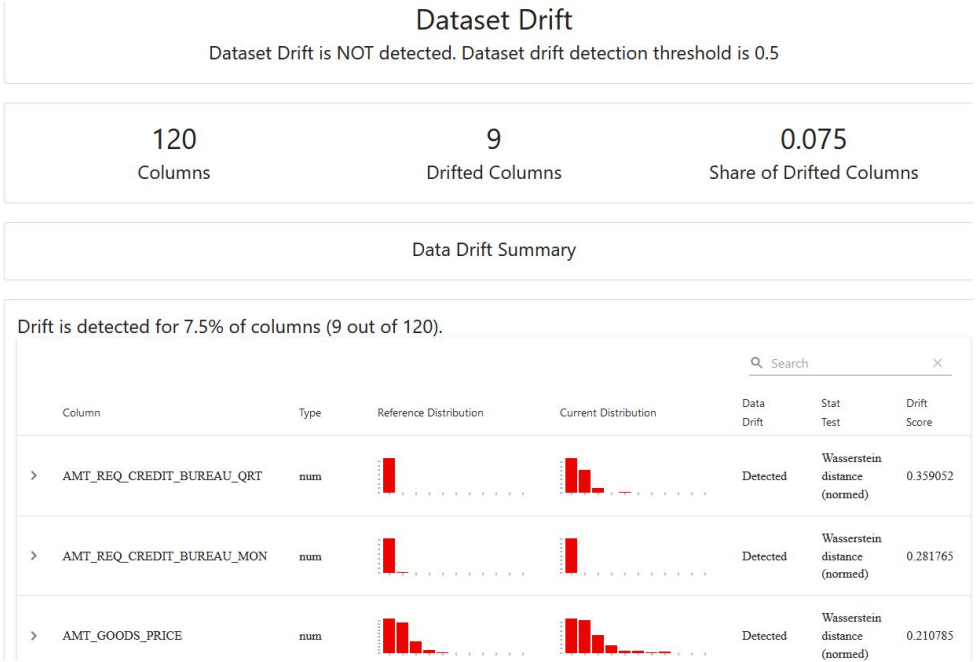
Heroku



Analyse du data drift

- Le data drift survient lorsque la distribution des données utilisées pour l'entraînement du modèle est statistiquement différente de la distribution des données en production.
- Le data drift impacte la performance du modèle.
- Surveillance des données avec EVIDENTLY AI
  - Données de référence : application_train
  - Données de production : application_test
  - Drift détecté sur 9 colonnes sur 120 (7,5%).
  - Pas de data drift détecté sur nos données.

- Cette surveillance des données en production permet de ré-entraîner le modèle si nécessaire, et ainsi de maintenir sa précision et sa fiabilité au fil du temps.



Dataset Drift

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

120	9	0.075
Columns	Drifted Columns	Share of Drifted Columns

Data Drift Summary

Drift is detected for 7.5% of columns (9 out of 120).

Search

Column	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

Extrait du rapport html de data drift obtenu à l'aide de EVIDENTLY AI - Rapport DataDriftPreset()

Conclusion

- Implémentation d'un modèle d'apprentissage supervisé adapté à un problème de classification binaire.
- Utilisation d'une fonction de coût métier
- Mise en œuvre d'une démarche MLOps avec MLflow.
- Utilisation d'un outil de gestion de versions de code (Git/GitHub).
- Déploiement du modèle de prédiction à l'aide d'une API (Flask API).
- Réalisation d'un dashboard (à l'aide de Streamlit) faisant appel à l'API via des requêtes.
- Déploiement de l'API et du dashboard dans le cloud à l'aide de Heroku.
- Mise en œuvre d'une stratégie de suivi de la performance du modèle dans le temps avec une analyse du data drift à l'aide de Evidently AI.

Merci pour votre attention

Questions ?