Documentation Technique

Application Pipeline Machine Learning

1- Présentation Générale de l'application

L'application Pipeline Machine Learning est comme son nom l'indique un Pipeline de Data Science avec entraînement, évaluation et prédiction de modèles de Machine Learning.

- **Principales Technologies utilisées** :

  - Python
  - Bibliothèques :
    - Streamlit pour le front,
    - Pandas et Numpy pour la manipulation des données
    - Scikit-learn, Lazypredict, pour la partie Machine Learning
    - Seaborn, Matplotlib pour la visualisation
    - Openpyxl, CSV, FPDF pour importer et exporter des fichiers
  - Plateforme d'hébergement : Streamlit Cloud

2- Architecture de l'application

main.py	Page principale de navigation
config.py	Fichier de style
routes.py	Fonctions de redirection
pages /	
0_Accueil.py	Page d'accueil de l'application
1_Import.py	Importation des données
2_Exploration.py	Analyse exploratoire + traitement
3_Machine Learning.py	Entraînement + sélection modèle
4_Evaluation.py	Évaluation modèle
5_Predictions.py	Prédictions manuelles / fichier
modules /	
module_eval.py	Fonctions d'évaluation des modèles
module_explo.py	Fonctions d'exploration et traitement
module_import.py	Fonctions d'import des fichiers
module_ML.py	Fonctions de machine learning
module_predict.py	Fonctions de prédiction

3- Installation et lancement

Pour récupérer le repo : git clone https://github.com/mon-utilisateur/Projet_IA_Vin.git

Pour installer les librairies : pip install -r requirements.txt

Pour lancer l'application : streamlit run main.py

L'application est également disponible via ce lien : https://pipeline-machine-learning.streamlit.app/

Si nécessaire appuyer sur le bouton pour « réveiller » l'application

4- Déroulement de votre visite sur l'application

- **Page 0** – Accueil :
  - Présentation générale de l'application
- **Page 1** – Import des données :
  - chargement CSV ou XLSX ou utilisation du dataset proposé,
  - suppression des colonnes inutiles
- **Page 2** – Exploration et Traitements :
  - Choix de la cible
  - Détection et choix du type de tâche (classification ou régression)
  - Visualisation de la distribution des colonnes
  - Encodage,
  - Analyse des corrélations et choix des colonnes à conserver
  - Gestion des NaN,
  - Gestion des outliers,
  - Standardisation
  - Export des données nettoyées en CSV ou XLSX
  - Téléchargement d'un rapport PDF des analyses et traitements effectués
- **Page 3** – Entraînement du modèle :
  - Séparation du jeu de données en un jeu d'entraînement et un jeu de test,
  - sélection automatique du meilleur modèle (LazyPredict / Cross-Validation),
  - Entraînement et export du modèle au format pickles
  - Optimisation des Hyperparamètres (GridSearchCV/RandomizedSearchCV) et nouvel export du modèle optimisé au format pickles
- **Page 4** – Évaluation :
  - rapport de classification / régression
- **Page 5** – Prédictions :
  - Import de nouvelles données par saisie manuelle ou import CSV/XLSX
  - Utilisation du modèle en mémoire ou import d'un autre modèle au format pickles pour effectuer des prédictions sur les nouvelles données

5- Points d’attention

- Bien valider **df_clean** pour que l’entraînement fonctionne
- Vérifier le bon encodage des cibles pour les prédictions
- **LabelEncoder** doit être utilisé si on veut faire le décodage lors de la prédiction

6- Points d’amélioration en vue d’une V.2

- Laisser l’utilisateur choisir le modèle de son choix malgré la recommandation
- Intégrer un réseau de neurones
- Intégrer du renforcement