1. Contrôle des documents

- **Titre**: Projet_ML_2023_D04
- **Version**: 1
- **Auteur(s)**: Groupe 2 - Benjamin BOUTE, Melody MIGLIACCIO, Vincent LOREAU, Adam OURRAD
- **Chef de projet**: Christophe GERMAIN
- **Date du rendu**: 26/11/2024

Lien vers l'application 😀
https://xkc7vpmzqitowvkchesrdm.streamlit.app/

2. Introduction

2.1 Objectif(s)

- Ce document a pour but de décrire toutes les technicités et l'architecture de l'application.

2.2 Champ(s) d'application

Cette application permet de :

- Faire une analyse descriptive du dataframe obtenu à partir du .csv chargé
- Des graphiques de distribution et pairplot
- Calcule la corrélation avec la target
- Calculer les fréquences
- Standardiser
- Gérer les valeurs manquantes

3. Vue d'ensemble du système

3.1 L'architecture du système

L'application est composée du dossier racine, d'un dossier "Data" et d'un dossier ".streamlit".

Dans la racine on trouve :

- Les différents scripts python pour chaque onglet ainsi que la fonction permettant de charger un fichier à traiter. Le script **"app.py"** gère les appels à ces différents fichiers.
- **requirements.txt** qui contient les prérequis pour le déploiement de l'application
- setup.sh
- **Procfile** qui permet de lancer l'application avec la bonne commande prompt.

Dans le dossier "Data" :

- **vin.csv** qui est le jeu de données utilisé pour la création de l'application

Dans le dossier ".streamlit" :

- **config.toml** qui est le fichier correspondant au thème d'affichage de l'application. Il peut être modifié.

a. Les fichiers .py

- **app.py :** Ce script gère les appels aux différentes composantes de l'application, la mise en forme du site et sert de page d'accueil.
- **general_analysis.py :** Calcule et affiche dans l'onglet correspondant les distributions, fréquences, boxplot, courbes de densité d'une variable choisie par l'utilisateur grâce à une selectbox parmi celles présentes dans le fichier .csv chargé. Permet aussi de calculer et afficher la matrice de corrélation.
- **machine_learning.py** : Met en place un pipeline de machine learning et permet de visualiser les résultats et prédictions. Le script permet de traiter les problèmes de classification et de régression en fonction du type de donnée que l'utilisateur a choisi. Il se base sur 6 algorithmes, 3 pour la classification et 3 pour la régression : Linear Regression, Random Forest Regressor et Support Vector Regression (SVR), Logistic Regression, Random Forest Classifier, Support Vector Machine (SVM)
  L'utilisateur est libre de choisir celui qu'il souhaite via une **selectbox**. Suite à l'entraînement , un fichier "model.pkl" sera créé et disponible au téléchargement pour l'utilisateur.
- **modelisation.py** : Calcule et affiche dans l'onglet correspondant un aperçu des données chargées, des statistiques de base (describe) et permet la gestion des valeurs manquantes.
- **preexistant.py** : Permet de charger un modèle .pkl déjà entraîné et de faire des prédictions sur une variable choisie par l'utilisateur avec celui-ci.

3.2 Les librairies

Le langage utilisé est Python. Nous utilisons le framework Streamlit pour créer la structure de l'application et gérer le déploiement.

Librairies basiques

- **numpy** : Fournit des outils puissants pour manipuler des tableaux et effectuer des calculs mathématiques efficaces.
- **Streamlit** : Permet de créer facilement des applications web interactives pour visualiser des données et des modèles.
- **pandas** : Outils pour la manipulation et l'analyse de données structurées sous forme de tableaux (DataFrames).
- **joblib** : Facilite la sauvegarde et le chargement d'objets Python volumineux, comme des modèles ou des données.
- **io** : Gère les flux d'entrée et de sortie, comme la lecture et l'écriture de fichiers (gère l'enregistrement du modèle en .pkl)

Librairies de visualisation

- **matplotlib.pyplot** : Permet de créer des graphiques statiques, simples et personnalisables en Python.
- **plotly.express** : Gère la création de graphiques interactifs avec un code minimal.
- **seaborn** : Ajoute des outils pour visualiser des données statistiques de manière élégante et informative.

Librairies pour le Machine Learning

Modules de Scikit-learn (Sklearn)

a. **Modules de sélections et de preprocessing :**

- **model_selection.train_test_split** : Sépare les données en ensembles d'entraînement et de test nécessaire pour créer notre IA.
- **preprocessing.StandardScaler** : Normalise les données en leur donnant une moyenne de 0 et un écart-type de 1.
- **preprocessing.OneHotEncoder** : Convertit des catégories en variables indicatrices (one-hot).
- **preprocessing.label_binarize** : Transforme des labels catégoriels en une représentation binaire pour des tâches multi-classes.

b. **Modules modélisations des modèles de régression et de classification :**

- **linear_model.LogisticRegression** : Implémente la régression logistique pour les classifications binaires et multi-classes.
- **linear_model.LinearRegression** : Modèle de régression linéaire pour des prédictions continues.
- **ensemble.RandomForestClassifier** : Classifie les données à l'aide d'un ensemble d'arbres de décision (forêts aléatoires).
- **ensemble.RandomForestRegressor** : Effectue des prédictions continues en utilisant des forêts aléatoires.
- **svm.SVC** : Implémente des machines à vecteurs de support (SVM) pour la classification.
- **svm.SVR** : Utilise les SVM pour effectuer des prédictions continues (régression).

c. **Modules évaluations des modèles :**

- **metrics.accuracy_score** : Évalue la précision d'un modèle en classification.
- **metrics.mean_squared_error** : Calcule l'erreur quadratique moyenne pour un modèle de régression.
- **metrics.confusion_matrix** : Génère une matrice de confusion pour analyser les prédictions (Les différences entre les vrais versus les prédictions du modèle)

d. **Autres modèles**

- **pipeline.Pipeline** : Facilite l'enchaînement d'étapes de traitement et d'apprentissage dans un flux structuré.
- **impute.SimpleImputer** : Remplace les valeurs manquantes par des valeurs statistiques (moyenne, médiane, etc.).
- **compose.ColumnTransformer** : Applique des transformations spécifiques à différentes colonnes d'un tableau de données.

4. Utilisation et sécurité

4.1. Utilisation

- L'application reste disponible tant que le déploiement reste actif.

4.2 Manuel opérationnel

- Instructions pour les opérations quotidiennes.
- https://github.com/VincentLo44/Digi_projet_ML_IA/blob/main/Documentation/Modop_visuel.pdf
- Les problèmes courants et leurs résolutions.

4.3. Sécurité

Mesures de sécurité

- Sécurité géré par Github et Streamlit

Contrôle d'accès

- Suivre le lien Streamlit pour se servir de l'application.
- **Attention : Il faut lier son compte Github à Streamlit pour redéployer régulièrement.**

5. Améliorations futures

- Faire une sécurité avec identification de compte
- Amélioration des modèles de machine learning
- Intégration de la courbe ROC (intégrer dans le code mais non fonctionnelle en l'état)

6. Annexe

- https://github.com/VincentLo44/Digi_projet_ML_IA