Mini-projet :
Prédiction du revenu annuel d'un marocain

1. Objectif

L'objectif général de ce mini-projet est de construire un pipeline complet de Machine Learning en Python pour prédire le **revenu annuel des Marocains** à partir de données simulées réalistes.

Le projet couvre toutes les étapes de développement d'un modèle de de Machine Learning à savoir : compréhension des données, préparation des données, modélisation, évaluation et déploiement.

2. Organisation et déroulement

- Les présentations des mini-projets auront lieu le mercredi **14 mai 2025**.
- Le mini-projet doit être réalisé sous forme d'un **Notebook** (.ipynb) sous **Jupyter Notebook** ou tout autre plateforme équivalente.
- Le notebook doit être bien documenté, avec des cellules Markdown expliquant chaque étape. Ajoutez des titres, des couleurs, des graphiques, etc., pour rendre la présentation plus claire.

3. Compétences visées

- Génération de données synthétiques à partir de contraintes statistiques
- Préparation des données
- Construction de pipelines avec `make_pipeline`
- Application de plusieurs modèles de régression
- Ajustement des hyperparamètres avec `GridSearchCV` ou `RandomizedSearchCV`

- Test des modèles
- Déploiement web du modèle prédictif avec **Streamlit**

4. Ensemble des données (Data Set)

En créant un script Python (generate_dataset.py), vous devez générer un dataset (dataset_revenu_marocains.csv) contenant environ 40000 enregistrements, en respectant les contraintes ci-dessous déduites à partir des statistiques du HCP (Haut-Commissariat au Plan) :

- Revenu moyen : **21.949 DH/an**
  - Urbain : 26.988 DH / Rural : 12.862 DH
- Répartition des revenus : 71,8% < moyenne (urbain 65,9%, rural 85,4%)
- Le revenu annuel varie en fonction de plusieurs facteurs, en particulier :
  - Âge : le revenu a tendance d'augmenter lorsqu'on se rapproche de l'âge de retraite.
  - Catégorie d'âge : jeune, adulte, sénior et âgé
  - Urbain/Rural : les personnes issues du milieu urbain ont tendance à gagner plus que les personnes du milieu rural.
  - Sexe : le revenu moyen des hommes est plus élevé que celui des femmes.
  - niveau d'éducation : (quatre niveaux à distinguer : sans niveau, fondamental, secondaire, supérieur), avec un revenu moyen plus élevé pour les personnes ayant un niveau d'enseignement supérieur.
  - Années d'expérience : Plus on est expérimenté, plus le revenu est élevé.
  - État matrimonial : Célibataire, marié, divorcé, veuf, divorcé
  - Possession de biens : (voiture, logement, terrain, etc.) ça peut refléter le niveau socio-économique d'un citoyen.

- Catégorie socioprofessionnelle : on peut distinguer 6 grands groupes (classés du plus haut revenu au plus bas) :
  1. Groupe 1 : est celui des responsables hiérarchiques de la fonction publique, directeurs et cadres de direction d'entreprises, cadres supérieurs et membres des professions libérales
  2. Groupe 2 : englobe les cadres moyens, des employés et des commerçants et intermédiaires commerciaux et financiers
  3. Groupe 3 : comprend les inactifs (retraités, rentiers, et autres inactifs)
  4. Groupe 4 : est celui des exploitants agricoles, pêcheurs, forestiers, chasseurs et travailleurs assimilés et les ouvriers agricoles
  5. Groupe 5 : comporte les conducteurs d'installations et de machines, les artisans et les ouvriers qualifiés
  6. Groupe 6 : inclue les manœuvres non agricoles, les manutentionnaires et travailleurs des petits métiers et les chômeurs n'ayant jamais travaillé.

- En plus des caractéristiques mentionné ci-dessus, vous devez ajouter au moins trois autres caractéristiques pertinentes de votre choix.
- Le dataset doit également inclure :
  - Valeurs manquantes
  - Valeurs aberrantes
  - Colonnes redondantes
  - Colonnes non pertinentes

5. Spécifications techniques

Le mini-projet doit être réalisé en respectant les spécifications techniques suivantes :

- Langage de programmation : **Python**

- Environnement de développement : **Jupyter Notebook** ou autre plateforme équivalente
- Packages du calcul scientifique et manipulation des données **: numpy** et **pandas**
- Package de visualisation des données : **matplotlib**, **seaborn**, **Sweetviz** ou autre.
- Package d'apprentissage automatique (machine learning) : **scikit-learn**.
- Déploiement web : **Streamlit** ou équivalent.
- Et tout autre librairie ou outil que vous estimez utile pour le développement du mini-projet.

6. Spécifications fonctionnelles

Au cours de ce mini-projet vous allez aborder l'ensemble des étapes du processus de Machine Learning.

Alors, voici l'ensemble des exigences que vous devez satisfaire en réalisant ce mini-projet.

6.1. Compréhension des données :

- **Chargement** du dataset
- **Affichage des données** : afficher un aperçu des 10 premières instances du dataset.
- **Description des données** : Afficher le **volume** (nombre total d'instances) et la **dimension** des données (nombre total des attributs), le **type** et le codage des attributs.
  Afficher les **statistiques descriptives** (moyenne, écart-type, quartiles, valeur minimale, valeur maximale, etc.). **Analyser** et **interpréter** les différentes valeurs.
- **Exploration des données** : afin d'approfondir votre compréhension des données et chercher d'éventuelles **corrélations** entre les variables du dataset, ainsi que de vérifier la qualité des données, vous devez réaliser une **analyse approfondie** en utilisant l'une des deux librairies **Sweetviz** ou **Pandas Profiling** (ydata-profiling).

N.B : Si vous avez constaté des incohérences (le dataset est déséquilibré, biaisé ou tout autre type d'anomalies) au niveau de la répartition des données du dataset, il faut revoir l'étape de génération des données.

N.B : les étapes de nettoyage, de transformation et de modélisation doivent être réalisées en créant des pipelines (en utilisant make_pipeline)

6.2. Nettoyage des données

Suite aux résultats d'analyse exploratoire des données que vous avez effectué dans l'étape précédente :

- Éliminer les éventuels **doublons** au niveau des **enregistrements**.
- Traitement des **valeurs manquantes** : proposer puis appliquer une technique de traitement des valeurs manquantes.
- Traitement des **valeurs aberrantes** : proposer puis appliquer une technique de traitement des valeurs aberrantes.

6.3. Transformation des données

- En fonction de votre compréhension des données, et **si nécessaire**, proposez (avec justification) la **suppression** des **attributs prédictifs** que vous estimez **non discriminants** (non pertinents). Ensuite, afficher un aperçu des données après l'application de cette transformation, le cas échéant.

- En fonction de votre compréhension des données, et **si nécessaire**, proposer (avec justification) la **création** de **nouveaux attributs prédictifs**. Ensuite, afficher un aperçu des données après l'application de cette transformation, le cas échéant.

- **Normaliser**, **si nécessaire** (avec justification), les valeurs des **attributs** prédictifs **numériques**. Ensuite, afficher un aperçu des données après l'application de la normalisation.

- Appliquez (avec justification) les **transformations** nécessaires pour convertir les colonnes de type **catégoriel** en type **numérique**. Ensuite, afficher un aperçu des données après l'application de ces transformations.

6.4. Séparation des données

- Diviser les données en deux sous-ensembles :
  - **Sous-ensemble d'apprentissage** : constitué de **70%** des lignes du dataset initial.
  - **Sous-ensemble de test** : constitué de **30%** des lignes du dataset initial.

6.5. Création et validation des modèles

Après les étapes de compréhension et de préparation des données, il est temps de passer à l'étape de **modélisation**.

L'objectif principal de cette étape est de créer **cinq modèles** de régression, à savoir, la **Regression Linéaire (LinearRegression)**, les **Arbres de Décision (DecisionTreeRegressor)**, les **Forêts d'Arbres Décisionnels (RandomForestRegressor)**, le **Gradient Boosting (GradientBoostingRegressor)** et les **Réseaux de neurones multi-couches (MLPRegressor)**. Ensuite, vous devez comparer et choisir bien évidemment celui qui donne les meilleures performances.

Dans ce mini-projet nous considérons : **MAE**, **RMSE**, ainsi que **R^2** comme étant les **mesures** de test des **performances**.

Pour atteindre cet objectif, vous devez :

- En utilisant la technique de **validation croisée, ajuster** (fine-tune) les **hyperparamètres** des cinq modèles de régression.

  Voici les hyperparamètres à ajuster :
  - **Régression linéaire** : *aucun* ;

- **Arbres de décision** :
  - ***criterion:*** ['squared_error', 'absolute_error'];
  - ***max_depth***: [None, 5, 6, 7, 10];
  - ***min_samples_split***: [2, 3, 4, 5, 10];
- **Forêts d'Arbres Décisionnels** :
  - ***n_estimators****:* [50, 100, 150, 200];
  - ***criterion:*** ['squared_error', 'absolute_error'];
  - ***max_depth****:* [None, 5, 10, 15, 20];
- **Gradient Boosting** :
  - ***loss:*** ['squared_error', 'absolute_error'];
  - ***learning_rate****:* [0.01, 0.1, 0.2];
  - ***n_estimators****:* [100, 200, 300];
  - ***subsample****:* [0.5, 0.8, 1];
- **Réseaux de neurones multi-couches** :
  - ***hidden_layer_sizes****:* [(50,), (100,), (100, 50), (100, 100)];
  - ***activation****:* ['relu', 'tanh', 'logistic'];
  - ***solver****:* ['adam', 'sgd'];
  - ***alpha****:* [0.0001, 0.001, 0.01];
  - ***learning_rate****:* ['constant', 'invscaling', 'adaptive'];
  - ***learning_rate_init****:* [0.001, 0.01, 0.1];
  - ***max_iter****:* [100, 200, 300];

- Après ajustement des hyperparamètres, comparer les performances des différents modèles. Quel est le modèle le plus performant ? Et quelles sont les valeurs de ses hyperparamètres ?
- Après avoir terminé le fine-tuning (ajustement) des hyperparamètres, utiliser le sous-ensemble d'apprentissage pour construire un modèle de régression en

employant l'algorithme et les hyperparamètres qui ont présenté les meilleurs résultats de performances durant la phase de validation.

6.6. Test du modèle

Selon les résultats obtenus dans la phase précédente, appliquer le modèle de régression sélectionné sur l'ensemble de **test** et afficher ses mesures de performances.
Si vous êtes satisfait par les résultats obtenus, passer au déploiement, sinon il faut revenir aux étapes précédentes (à partir de l'étape de création du dataset jusqu'à l'étape de modélisation) et essayer de faire les modifications nécessaires.

6.7. Déploiement

- **Sauvegarder** le **modèle** sélectionné (en utilisant la librairie **joblib**)
- Créer une **API** pour le modèle sauvegardé en utilisant **FastAPI**.
- Créer une **application web** simple via **Streamlit** ou autre à travers laquelle vous faites appel à l'API FastAPI.
  - Interface utilisateur pour saisir les valeurs des différentes variables prédictives.
  - Affichage de la prédiction du **revenu annuel**.
  - Vous pouvez proposer d'autres fonctionnalités

7. Livrables :

- Script Python de génération des données : generate_dataset.py
- Fichier CSV du dataset : dataset_revenu_marocains.csv
- Notebook du mini-projet : mini_projet_AI_Noms.ipynb
- Sauvegarde du modèle sélectionné : modele_selection.joblib
- API FastAPI : api.py
- Application web : app.py

- Un fichier README.md
- Tout autre fichier nécessaire pour le bon fonctionnement du notebook ou de l'application web de déploiement.