# **Documentation et Explication du Code**

## **Description Générale**

Ce script gère l'exécution asynchrone de tâches de traitement de données à l'aide de la bibliothèque **APScheduler**. Il est conçu pour être intégré dans une application qui permet de planifier, exécuter, et suivre l'état de tâches de traitement de données de manière asynchrone.

## **Composants Clés**

### **1. TaskStatus**

* **Objectif** : Cette classe énumère les différents états qu'une tâche peut avoir au cours de son cycle de vie.
* **États** :
  + WAITING : La tâche est en attente de démarrage.
  + IN\_PROGRESS : La tâche est en cours d'exécution.
  + DONE : La tâche est terminée avec succès.
  + ERROR : Une erreur s'est produite lors de l'exécution de la tâche.

### **2. global\_store**

* **Objectif** : Ce dictionnaire global stocke les objets et les données nécessaires à la prédiction, comme le modèle, le chemin du scaler, les données prétraitées, les noms des caractéristiques, et la variable dépendante.

### **3. process\_task**

* **Objectif** : Cette fonction asynchrone exécute le traitement de la tâche. Elle prend en charge le prétraitement des données, l'entraînement du modèle, et la génération de résultats à stocker dans la tâche.
* **Paramètres** :
  + task\_id : Identifiant unique de la tâche.
  + name : Nom de la variable cible pour la prédiction.
  + file : Fichier CSV à traiter.
  + f : Contenu du fichier en tant qu'objet de type bytes.
  + scale : Indicateur pour savoir si les données doivent être mises à l'échelle.
* **Processus** :
  + Convertit le contenu binaire en un objet de type fichier.
  + Appelle la fonction preprocess\_data\_pipeline pour prétraiter les données et entraîner le modèle.
  + Stocke le modèle, le scaler, et les données prétraitées dans global\_store.
  + Génère un DataFrame avec les données prétraitées et une visualisation des prévisions effectuées par le modèle Prophet.
  + Enregistre le statut de la tâche (DONE ou ERROR) et le résultat ou l'erreur associée.
* **Sortie** :
  + Aucune sortie directe ; la fonction met à jour l'état de la tâche dans le dictionnaire global tasks.

### **4. schedule\_task**

* **Objectif** : Cette fonction planifie l'exécution d'une tâche asynchrone en utilisant **APScheduler**.
* **Paramètres** :
  + task\_id : Identifiant unique de la tâche.
  + name : Nom de la variable cible pour la prédiction.
  + file : Fichier CSV à traiter.
  + f : Contenu du fichier en tant qu'objet de type bytes.
  + scale : Indicateur pour savoir si les données doivent être mises à l'échelle.
* **Processus** :
  + La tâche est planifiée pour s'exécuter après un délai de 1 seconde.
  + APScheduler gère l'exécution de la fonction process\_task à la date et heure spécifiées.

### **5. scheduler**

* **Objectif** : Un instance de **AsyncIOScheduler** est utilisée pour gérer la planification des tâches.
* **Fonctionnement** : Il est démarré en fin de script et reste actif pour gérer les tâches planifiées.

## **Résumé**

Ce script est conçu pour gérer de manière asynchrone le prétraitement des données et l'entraînement de modèles de prédiction avec des fonctionnalités avancées de suivi des tâches. Il utilise APScheduler pour la planification des tâches et stocke les résultats et les états dans une structure de données centralisée pour faciliter le suivi et la gestion des tâches.