**PIPELINE TRAITEMENT DES DONNE ET NETOYAGE**

**Pipeline de Nettoyage des Données : Explication Pas à Pas**

1. **Importation des bibliothèques et du jeu de données**
   * **But** : Charger les bibliothèques nécessaires (pandas, numpy, etc.) et lire le fichier CSV contenant les avis d’utilisateurs Uber.
   * **Code exemple** :
   * import pandas as pd
   * df = pd.read\_csv("uber\_data.csv")
   * **Pourquoi** : Pour disposer des données dans un DataFrame Pandas, ce qui facilite le nettoyage et l’analyse.
2. **Fusion de reviewCreatedVersion et appVersion**
   * **But** : Certaines lignes n’avaient pas de valeur dans reviewCreatedVersion mais en avaient une dans appVersion, et inversement. On décide de fusionner ces informations.
   * **Code exemple** :
   * df['reviewCreatedVersion'] = df['reviewCreatedVersion'].fillna(df['appVersion'])
   * **Pourquoi** : Ne pas perdre l’information « version de l’application » : si reviewCreatedVersion est vide, on la remplace par appVersion.
3. **Suppression des colonnes redondantes ou inutiles**
   * **Colonnes supprimées** :
     + appVersion : devient redondante une fois la fusion réalisée.
     + userImage, replyContent, repliedAt : soit parce qu’elles sont vides, soit parce qu’on n’en a pas besoin pour la suite de l’analyse.
   * **Code exemple** :
   * df.drop(columns=['appVersion', 'userImage', 'replyContent', 'repliedAt'], inplace=True)
   * **Pourquoi** : Garder un DataFrame plus léger et plus clair en ne conservant que les colonnes réellement utiles pour notre analyse ou nos futurs KPI.
4. **Gestion des types de données (conversion)**
   * **Colonne score** : convertie en entier (int), parfois en remplaçant les éventuelles valeurs manquantes par 0.
   * **Colonne thumbsUpCount** : convertie également en entier.
   * **Code exemple** :
   * df['score'] = df['score'].fillna(0).astype(int)
   * df['thumbsUpCount'] = df['thumbsUpCount'].fillna(0).astype(int)
   * **Pourquoi** : Les notes (score) et le nombre de « likes » (thumbsUpCount) sont numériquement plus simples à manipuler en tant qu’entiers (calculs de moyennes, histogrammes, etc.).
5. **Conversion des dates**
   * **But** : Convertir les colonnes de date/heure (at) en format datetime afin de pouvoir réaliser des analyses temporelles (groupby par jour, mois, etc.).
   * **Code exemple** :
   * df['at'] = pd.to\_datetime(df['at'], errors='coerce')
   * **Pourquoi** : Le format datetime permet de trier, filtrer ou grouper plus facilement les données sur l’échelle du temps (par exemple, comparer le nombre de reviews par mois).
6. **Suppression des lignes avec valeurs manquantes**
   * **But** : Après les conversions et la fusion, il est possible qu’il reste des lignes incomplètes (valeurs nulles ou NaN) qu’on souhaite exclure.
   * **Code exemple** :
   * df.dropna(inplace=True)
   * **Pourquoi** : On préfère parfois avoir moins de données mais 100% propres, plutôt que conserver des lignes incomplètes qui pourraient fausser les analyses.
     + *Attention* toutefois : cela dépend du taux de perte. Il faut s’assurer que ce n’est pas trop drastique.
7. **Enregistrement des données nettoyées**
   * **But** : Sauvegarder la version finale « propre » du DataFrame pour ne pas avoir à refaire tout le pipeline.
   * **Code exemple** :
   * df.to\_csv('uber\_data\_cleaned.csv', index=False)
   * **Pourquoi** : Les transformations effectuées en mémoire ne sont pas permanentes ; il faut un nouveau fichier de sortie pour réutiliser ce jeu de données nettoyé facilement.