# Document Technique - API de Prétraitement et Prédiction de Sentiments

## 1. Introduction

Ce projet consiste à développer une API FastAPI qui nettoie et prédit la classe de sentiment de commentaires de films en utilisant un modèle de classification Bernoulli Naïve Bayes. L'application web utilise Streamlit pour permettre aux utilisateurs de soumettre des fichiers CSV ou de saisir des commentaires manuellement, qui sont ensuite traités par l'API. Le modèle prédit ensuite une classe de sentiment sur une échelle de cinq niveaux.

## 2. Structure des Fichiers

### 2.1. Fichiers Principaux

- **bernoulli_model.ipynb** : Notebook contenant le modèle Bernoulli Naïve Bayes, utilisé pour entraîner et évaluer les données avant son déploiement en API.
- **preprocessing.py** : Script contenant toutes les fonctions de prétraitement nécessaires au nettoyage des données avant la prédiction.
- **api.py** : Script qui définit l'API FastAPI, permettant de recevoir des fichiers ou des textes et de renvoyer des prédictions après nettoyage.
- **app.py** : Application Streamlit, qui sert d'interface utilisateur pour soumettre des données à l'API.

## 3. API FastAPI

### 3.1. Endpoints

- `/predict` : Prend en entrée un fichier CSV ou du texte manuel, nettoie les données et renvoie la classe de sentiment prédite.
- `/clean-csv` : Nettoie un fichier CSV/TSV et renvoie les données nettoyées sans effectuer de prédiction.
- **Exigences techniques** :
    - Le fichier CSV/TSV doit avoir un séparateur '\t'.
    - Pour les entrées manuelles, le texte doit comporter un minimum de 50 caractères et être en anglais.

### 3.2. Fonctionnement

L'API est configurée pour accepter deux types de requêtes : une requête d'analyse de fichier CSV/TSV et une requête de prédiction basée sur un texte manuel. Le texte passe d'abord par des étapes de nettoyage (tokenisation, suppression des mots vides, lemmatisation, etc.) avant que le modèle Bernoulli effectue une prédiction.

# 4. Script de Prétraitement (`preprocessing.py`)

## 4.1. Fonctions de Prétraitement

- **replace_empty_with_nan** : Remplace les valeurs vides par des NaN.
- **remove_stopwords** : Supprime les mots vides de la phrase.
- **lemmatize_tokens** : Applique la lemmatisation aux tokens pour normaliser les mots.
- **remove_duplicates_by_column** : Supprime les doublons dans les colonnes spécifiées.
- **remove_consonant_or_vowel_sequences_from_tokens** : Nettoie les séquences de voyelles ou de consonnes dans les tokens.

Ces fonctions sont appliquées séquentiellement pour préparer les données avant la prédiction par le modèle.

# 5. Application Streamlit

## 5.1. Fonctionnalités

L'application propose deux onglets pour l'utilisateur :

- **Upload CSV/TSV** : Permet de soumettre un fichier CSV/TSV qui sera nettoyé et traité par l'API.
- **Texte Manuel** : Permet à l'utilisateur de soumettre un commentaire en anglais, qui sera nettoyé et passé au modèle pour une prédiction.

## 5.2. Prédictions et Affichage

- **Nettoyage** : Le fichier CSV/TSV ou le texte manuel est d'abord nettoyé, et un DataFrame est affiché à l'utilisateur.
- **Prédictions** : Les prédictions sont ensuite ajoutées sous forme de nouvelle colonne dans le DataFrame et affichées dans l'application Streamlit.

# 6. Modèle Bernoulli Naïve Bayes

Le modèle utilisé pour la prédiction des sentiments est un Bernoulli Naïve Bayes, qui est entraîné sur un dataset Rotten Tomatoes comportant des classes de sentiments échelonnées sur cinq niveaux. L'entraînement et l'évaluation sont suivis avec MLFlow.

## 6.1. Suivi des Paramètres avec MLFlow

- **Modèle** : Le modèle est sauvegardé sous forme de fichier `.joblib` pour une utilisation ultérieure dans l'API.
- **Hyperparamètres** : Les hyperparamètres sont enregistrés dans MLFlow, incluant le suivi de la performance du modèle.

# 7. Conclusion

L'ensemble de l'API et de l'application Streamlit est maintenant opérationnel, permettant aux utilisateurs de soumettre des fichiers CSV/TSV ou des textes pour une analyse de sentiment. Grâce à MLFlow, nous suivons les performances du modèle et pouvons ajuster les hyperparamètres de manière dynamique pour améliorer les résultats futurs.